P.1877/92

2 1992

informatyka

WYDAWNICTWO SIGMA NOT

Cena 15 000 zł

informatyka

nr 2 1992
Luty
Rok wydania XXVII

Miesięcznik
ISSN 0542-9951
INDEKS 36124

KOLEGIUM REDAKCYJNE:

mgr Jarosław DEMINET
mgr inż. Piotr FUGLEWICZ
mgr Teresa JABŁOŃSKA
(sekretarz redakcji)
Władysław KLEPACZ
(redaktor naczelny)
dr inż. Wojciech MOKRZYCKI
mgr inż. Jan RYŻKO
dr Zdzisław SZYJEWSKI
mgr Hanna WŁODARSKA

PRZEWODNICZĄCY
RADY PROGRAMOWEJ:

Prof. dr hab.
Juliusz Lech KULIKOWSKI

WYDAWCA:
Wydawnictwo Czasopism i Książek
Technicznych SIGMA NOT
Spółka z o.o.
ul. Biała 4
00-950 WARSZAWA
skrytka pocztowa 1004

Redakcja:
01-552 Warszawa,
Pl. Inwalidów 10, p. 104, 105
tel. 39-14-34

Materiałów nie zamówionych
redakcja nie zwraca

W sprawach ogłoszeń prosimy zwracać się bezpośrednio do Redakcji lub Działu Reklamy i Marketingu
00-950 Warszawa
ul. Biała 4
telefon: 20-31-24
telefaks: 20-31-16
teleks: 814550

W numerze:

Strona

Propozycja strategii rozwoju informatyki i jej zastosowań w Rzeczypospolitej Polskiej (2) 1
Scheduler transakcyjny dla zastosowań komunikacyjnych – *Zbigniew Fryźlewicz, Jędrzej Ułasiewicz* 10
Emulator układowy mikroprocesorów jako narzędzie uruchomieniowe w ujęciu programistycznym – *Dariusz Zonenberg* 15
LINC – kompletne środowisko CASE – *Roman Cupisz* 20
Strukturalna analiza i projektowanie systemów informatycznych – metodyka Gane'a – Sarsona – *Katarzyna Stąpor* 24

/unix
Integracja systemów operacyjnych MS-DOS i UNIX (2) – *Paweł Żak* 29

Listy
Szanowna Redakcjo! – *Jacek Przepióra* III okł.

W najbliższych numerach:

- Zakończenie druku ostatniej części „Propozycji rozwoju informatyki i jej zastosowań w Rzeczypospolitej Polskiej".
- Mariusz Borsa omawia typową architekturę systemu interfejsu użytkownika oraz dostępne na rynku produkty wspomagające programowanie takich interfejsów.
- Jan Baranowski charakteryzuje nowoczesne, wielopoziomowe środowisko baz danych firmy NOVELL.
- Katarzyna Stąpor zamyka tematykę strukturalnej analizy i projektowania systemów informatycznych, poruszoną w poprzednim numerze.
- Piotr Fuglewicz podaje kilka informacji o językoznawstwie statystycznym.

Warunki prenumeraty

Przyjęcie prenumeraty – wyłącznie na podstawie dokonanej wpłaty na drukach dostarczanych dotychczasowym prenumeratorom przez Wydawnictwo, lub nowym – po uprzednim zgłoszeniu zapotrzebowania (pisemnie lub telefonicznie) w Zakładzie Kolportażu Wydawnictwa.

Blankiet wpłaty – powinien zawierać następujące informacje: dokładna nazwa i adres (z kodem pocztowym) zamawiającego, tytuły zamawianych czasopism, ich liczbę i okres prenumeraty.

Wpłata – zgodnie z podanymi cenami należy dokonać w banku lub w UPT na konto podane na naszym blankiecie, tj:
Państwowy Bank Kredytowy III O/Warszawa nr: 370015-1573-139-11

Prenumeratorzy zbiorowi – osoby prawne obowiązują blankiety „Wpłata-Zamówienie". Cena normalna.

Prenumeratorzy indywidualni – osoby fizyczne obowiązują blankiety typu przekazy dla wpłat na rachunki bankowe. Cena normalna.

Prenumerata ulgowa – zgodnie z podaną ceną ulgową przysługuje wyłącznie osobom fizycznym, będącym członkami SNT, studentom i uczniom szkół zawodowych. Uczniowie szkół ogólnokształcących mogą zamówić w prenumeracie ulgowej tylko miesięcznik „Aura"
Uwaga! w podanym okresie prenumeraty można zamówić tylko po jednym egzemplarzu z każdego tytułu.

Prenumerata ze zleceniem wysyłki za granicę – cena prenumeraty ze zleceniem wysyłki za granicę jest dwukrotnie wyższa od ceny normalnej.
Należy podać dokładny adres odbiorcy za granicą.

Terminy przyjmowania prenumeraty:
– do 10 listopada na I, II, III, IV kwartał następnego roku
– do 28 lutego na II, III, IV kwartał br.
– do 31 maja na III i IV kwartał br.
– do 31 sierpnia na IV kwartał br.

Zmiany w prenumeracie, np. zmiana liczby tytułów, liczby egzemplarzy, rezygnacja z prenumeraty, można zgłaszać tylko w podanych terminach z mocą obowiązującą od następnego kwartału.

Egzemplarze archiwalne (z lat ubiegłych)
Można nabyć za gotówkę w Klubie Prasy Technicznej, Warszawa, ul. Mazowiecka 12 (tel. 26-80-16) lub zamówić pisemnie w Zakładzie Kolportażu, Dział Handlowy, 00-950 Warszawa, skr. poczt. 1004 (tel. 40-37-31), na rachunek lub za zaliczeniem pocztowym.

Informacji o prenumeracie udziela: Zakład Kolportażu Wydawnictwa SIGMA-NOT Spółka z o.o., 00-716 Warszawa, ul. Bartycka 20, skr. 1004. Telefony: 40-00-21 wewn. 293, 295, 299 lub 40-30-86, 40-35-89.

Wstępna cena jednego egzemplarza na 1992 rok: normalna – 15 000 zł ulgowa – 12 250 zł

Wartość prenumeraty (w zł):
Normalna: kwartalna – 45 000, półroczna 90 000, roczna 180 000
Ulgowa: kwartalna – 33 750, półroczna 67 500, roczna 135 000

Uwaga: W przypadku zmiany cen w okresie objętym prenumeratą, prenumeratorzy zobowiązani są do dopłaty różnicy cen.

Skład i druk: Drukarnia SIGMA NOT Sp. z o.o. z. 669/91

Propozycja strategii rozwoju informatyki i jej zastosowań w Rzeczypospolitej Polskiej (2)

P. 1877/92

Rozdział 5

Gospodarka zasobami informacyjnymi

W skład zasobów informacyjnych państwa wchodzą wszelkie informacje niezbędne do jego funkcjonowania, bez względu na sposób ich przechowywania. Do takich zasobów należą m.in. zbiory przepisów składających się na obowiązujący ustrój prawny państwa, wszelkie dane ewidencyjne (ludność, grunty, zasoby surowcowe, instytucje i podmioty gospodarcze itp.), dane gospodarcze oraz dane statystyczne. Dobre funkcjonowanie nowoczesnego państwa, jego funkcjonariuszy i jego instytucji, a także przedsiębiorstw gospodarczych i obywateli, wymaga szybkiego i łatwego dostępu do takich zasobów informacyjnych, oczywiście przy zachowaniu odpowiednich wymagań poprawności i ochrony danych, o czym piszemy w rozdz. 11.

Wysoko rozwinięte kraje Europy i Ameryki Północnej już dawno przystąpiły do budowy odpowiednich systemów informatycznych, przyjmując różne podejścia do udostępniania zasobów informacyjnych. Skrajne podejścia reprezentują tu: model USA, w którym znaczna część zasobów informacyjnych jest gromadzona i udostępniana przez instytucje prywatne (np. prywatne banki danych prawnych) oraz model francuski, w którym gospodarka zasobami informacyjnymi jest w dużej mierze pod kontrolą państwową.

W Polsce w ubiegłych latach podjęto próby tworzenia systemów informatycznych gospodarki zasobami informacyjnymi. Do najważniejszych należą: PESEL – system ewidencji ludności oraz REGON – system ewidencji podmiotów gospodarczych. Oba te systemy, w związku ze zmianą systemu politycznego i gospodarczego Polski, wymagają gruntownej reorganizacji. Systemy zbierania informacji gospodarczej przez Główny Urząd Statystyczny, nastawione wyłącznie na gospodarkę państwową, nie wychwytują prawidłowo tendencji gospodarczych w sektorze prywatnym. W latach siedemdziesiątych w kancelarii sejmu rozpoczęto prace nad systemem wyszukiwania przepisów prawnych, prace te zostały jednak wkrótce przerwane. Ostatnio obserwuje się też wiele lokalnych prób konstruowania systemów, takich jak ewidencja gruntów, pojazdów, bezrobotnych. Tego typu systemy są jednak niespójne i różnią się zasobami informacyjnymi. Nieśmiałe prace nad systemami informacyjnymi podejmują też organy samorządu terytorialnego. Prace te są jednak hamowane niedostatkiem środków finansowych.

Należy podkreślić, że wszystkie dotychczasowe informatyczne systemy informacyjne w Polsce były konstruowane wyłącznie w celu dostarczania informacji organom władzy państwowej. Taki stan uważamy za niedopuszczalny w nowym systemie politycznym i gospodarczym, gdzie **obywatele i inne podmioty prawne mają uzasadnione interesy w korzystaniu z zasobów informacyjnych państwa.** Nowe rozwiązania powinny stwarzać im możliwości dostępu do niezbędnych informacji na takich samych warunkach, jak organom władzy państwowej.

Część informacji dotychczas zbieranych przez państwo za pomocą środków informatycznych (i nie tylko) narusza prawa osobiste obywateli (casus: system PESEL). Nie uregulowane prawnie zasady dostępu do informacji i ich niemal swobodny wypływ poza granice kraju również mogą naruszać takie prawa. Stąd też do najważniejszych postulatów związanych z gospodarką zasobami informacyjnymi należy **szybkie podjęcie prac nad prawną ochroną dóbr obywateli w związku z rozwojem nowych technik informatycznych.**

Budowa nowoczesnych systemów informacyjnych wymaga budowy rozległych sieci komputerowych. Prace nad siecią pakietową POLPAK powinny być znacznie zintensyfikowane i skoordynowane z rozwojem systemów informacyjnych, o których piszemy w dalszym ciągu tego rozdziału. Sieciom komputerowym jest poświęcony rozdz. 7.

Inwestycje niezbędne do stworzenia właściwej infrastruktury gospodarki zasobami informacyjnymi są drogie i nie należy się spodziewać, aby w najbliższym czasie zostały podjęte przez podmioty pozarządowe. Prócz tego pewne elementy zasobów informacyjnych (wszelkie dane osobowe o obywatelach, szczegółowe informacje gospodarcze, dane ewidencyjne) nie mogą być powierzone w zarząd takim podmiotom ze względu na ich poufny i strategiczny charakter. Dodatkowo, trwający właśnie proces przemian struktury własnościowej gospodarki wymaga szybkiego udostępniania i aktualizowania danych ewidencyjnych z nim związanych. Wszystkie te argumenty wskazują, że państwo musi wziąć na siebie ciężar niezbędnych inwestycji.

Za najważniejsze w tym zakresie uważamy:

1. Prawne ustalenie wymagań funkcjonalnych, jakie powinny spełniać powszechne systemy informacyjne, zasad dostępu do zawartych w nich danych i ich ochrony. W tej sprawie można oprzeć się na doświadczeniach EWG.
2. Budowę państwowego systemu informatycznego z pełną informacją o ustroju prawnym państwa z możliwością zdalnego dostępu przez rozległą sieć komputerową (por. rozdz. 7). System ten powinien być dostępny dla organów państwowych, samorządowych, podmiotów gospodarczych i wszystkich obywateli. Koszty budowy takiego systemu można amortyzować wprowadzając odpłatność za korzystanie z jego usług.
3. Dostosowanie zakresu informacji zbieranych przez Główny Urząd Statystyczny do norm EWG. Powinno temu towarzyszyć prawne uregulowanie zasad dostępu do informacji gospodarczych i statystycznych przez pozapaństwowe podmioty prawne i obywateli oraz budowa odpowiedniego systemu informatycznego dostarczającego syntetyczne informacje gospodarcze. Istotną część takiego systemu powinien stanowić zbiór bilansów przedsiębiorstw gospodarczych zobowiązanych do ich publikowania. Również ten system powinien być powszechnie dostępny przez sieć komputerową.
4. Ustalenie norm funkcjonalnych jakie powinny spełniać gminne systemy informacyjne w odniesieniu do zadań zlecanych gminom przez administrację państwową. Częściowe finansowanie gminnych systemów informacyjnych.
5. Pełną i jednolitą informatyzację systemu ewidencji gruntów i ksiąg wieczystych. Koszty inwestycji związanej z takim systemem zostaną szybko zrekompensowane w związku z usprawnieniem przy jego pomocy procesu przemian własnościowych.

Uważamy, że wymienione tutaj inwestycje mogą być częściowo finansowane z kredytów zagranicznych przeznaczonych na restrukturyzację systemu gospodarczego Polski. Raz stworzo-

ne, i potem odpowiednio pielęgnowane, systemy informatyczne znacznie usprawnią zarządzanie państwem, a także podejmowanie decyzji przez obywateli i podmioty gospodarcze.

Prócz tego rząd powinien wspierać działania tych podmiotów pozarządowych, które zdecydują się na samodzielne inwestowanie w rozwój publicznych zasobów informacyjnych kraju. Pomoc rządu w tym zakresie powinna, naszym zdaniem, polegać na gwarancjach kredytowych na takie przedsięwzięcia, a także na odpowiedniej polityce stymulacyjnej, którą szerzej zajmiemy się w rozdz. 9. Wspierane projekty powinny być nadzorowane merytorycznie przez odpowiednie organa państwowe lub niezależne, upoważnione do tego instytucje, pod kątem spełnienia wymagań bezpieczeństwa i ochrony danych, a także praw obywatelskich.

Rozdział 6

Informatyzacja administracji państwowej

Komputeryzacja i informatyzacja

Zacznijmy od wyjaśnienia tylko z pozoru drobnej różnicy terminologicznej:
– **komputeryzacja** oznacza zakup sprzętu komputerowego, a dopiero potem tworzenie ad hoc systemów informatycznych;
– **informatyzacja** oznacza zaprojektowanie systemu informatycznego spełniającego zapotrzebowanie, a następnie wybranie i zakup odpowiedniego dla niego sprzętu komputerowego.
W rozdziale 4 tego opracowania wyraźnie podkreśla się przewagę informatyzacji nad komputeryzacją.

Wbrew ogólnie przyjmowanym poglądom, inwestowanie w informatyzację firmy lub instytucji nie przynosi bezpośrednio zysków. Z reguły jest to działalność czasochłonna, bardzo kosztowna i często połączona z dużym ryzykiem w uzyskaniu powodzenia. Pomimo tego istnieją jednak przesłanki powodujące szybki rozwój zastosowań informatyki – na świecie w makroskali, w Polsce na razie w mikroskali. Do najważniejszych motywacji informatyzacji należy dążenie do:
– uporządkowania struktury organizacyjnej zarządzania firmą lub instytucją;
– przyśpieszenia obsługi klientów przez stanowiska bezpośredniego dostępu;;
– zmniejszenia zatrudnienia w obsłudze przetwarzania dokumentów;
– unowocześnienia instytucji;
– dopasowania się do instytucji współpracujących, które są już zinformatyzowane;
– poszukiwania nowych metod podejmowania decyzji na podstawie analizy, a następnie syntezy już zgromadzonych informacji. Łatwo widzieć, że uzyskanie rzeczywistych efektów z zainstalowania systemu informatycznego można osiągnąć jedynie w drodze racjonalnego wybrania obszaru zastosowania informatyzacji. Konieczne jest więc włożenie znacznego wysiłku w określenie celów informatyzacji na początku każdego przedsięwzięcia i sposobów jego poprawnej realizacji, gdyż łatwo możemy osiągnąć stan przeciwny do zamierzonego – przynoszący wyliczalne straty.

Systemy informatyczne

Ocena obecnego stanu istniejących systemów informatycznych w administracji jest tutaj dokonana bardzo ogólnie, wykonanie szczegółowej ekspertyzy wymagałoby bowiem dostępu do wielu informacji rozproszonych w różnych urzędach. Ponadto występująca równocześnie reorganizacja struktur zarządzania powoduje, że wiele informacji się zdezaktualizowało i część systemów w ich dotychczasowym kształcie jest już niepotrzebna.

Praktycznie każdy resort eksploatuje lub opracowuje własny system informatyczny. Oprócz systemów centralnych PESEL, REGON i POLTAX istnieje wiele różnego typu systemów tworzonych lokalnie, takich jak ewidencja gruntów, pojazdów, bezrobotnych itp. Systemy te są wzajemnie niespójne, często przechowując te same informacje. Istnieje też wiele systemów autonomicznych na szczeblu lokalnym i centralnym, współpracujących (lub mających współpracować) z systemami administracji państwowej. Przykładem są systemy ubezpieczeń społecznych, obsługi banków, rozliczeń energii, finansowo-księgowe, kadrowe i płacowe, magazynowe oraz zarządzania produkcją.

Nie negując użyteczności tych wszystkich systemów w przeszłości, trzeba stwierdzić, że są one już dziś przestarzałe w swojej strukturze, metodach realizacji oraz za sprawą sprzętu, na którym funkcjonują (zużyte komputery Odra 130x, Riad, SM itp.). Z kolei nowe systemy są realizowane z wykorzystaniem prostych systemów zarządzania bazami danych, funkcjonujących na sprzęcie mikrokomputerowym, w części połączonym w sieci lokalne. Większość tych systemów zrealizowano bez stosowania metodyki inżynierii projektowania „od specyfikacji użytkowej do realizacji technicznej", często z wykorzystaniem nie licencjonowanych (czytaj: niepewnych) narzędzi programowych i bez odpowiedniej walidacji zgodności działania z oczekiwaniami użytkownika. Dużą wadą tych systemów jest też ich mała odporność na nieupoważniony dostęp oraz wirusy.

Jednocześnie konieczne jest uświadomienie sobie faktu marnotrawienia ogromnych (proporcjonalnie do aktualnych możliwości budżetowych) środków na dublowanie prac, z których w rezultacie żadna nie jest dobra (chociaż niektóre z nich mogą być jak na te warunki poprawne).

Infrastruktura informatyzacji w administracji

Wdrożenie systemu informatycznego w instytucji wymaga istnienia odpowiedniej infrastruktury, którą jest:
– miejsce i sposób instalacji sprzętu;
– kadra informatyczna;
– przygotowanie użytkowników.
Problem kadry informatycznej w administracji został przedstawiony w rozdz. 3. Tutaj trzeba podkreślić, że praktycznie przy każdym większym systemie musi istnieć zespół zajmujący się administracją systemu i pomagający użytkownikom (urzędnikom) w jego eksploatacji. Co więcej, zatrudnieni w zespole informatycy powinni mieć duże doświadczenie i dość szeroki zakres umiejętności. Obecnie wymagania te jest trudno pogodzić z oferowanymi w administracji wynagrodzeniami dla specjalistów. Trzeba tutaj założyć, że za dwa, trzy lata wartość programisty (dobrego) w Polsce będzie prawie równa wartości programisty na Zachodzie. Konieczne będzie też w najbliższym czasie zatrudnianie specjalistów zagranicznych dla wspomagania realizacji projektów.

Również niedoceniany jest problem właściwego szkolenia użytkowników takich systemów. Firmy oferujące szkolenia są słabo przygotowane merytorycznie. Kuriozalne są też szkolenia prowadzone przez firmy zagraniczne w obcym języku z tłumaczeniem na język polski. Szkolenie źle przygotowane i prowadzone bez doświadczeń dydaktycznych zmniejsza akceptację informatyzacji instytucji przez jej urzędników. W rezultacie spada efektywność wykorzystania systemu informatycznego. W drastycznych przypadkach może to prowadzić do całkowitej jego negacji, łącznie z dokonywaniem sabotażu.

Cechy systemów informatycznych

Wyróżniamy trzy podstawowe warstwy systemu informatycznego:
– **ewidencyjną,** z funkcjami gromadzenia, weryfikowania i porządkowania informacji wprowadzanych do systemu i w nim przetwarzanych;
– **informacyjną,** z funkcjami wyszukiwania określonych pojedynczych lub zbiorczych informacji istniejących w systemie oraz dostarczających zestawień według zadanych zapytań;
– **decyzyjną,** będącą systemem doradczym (eksperckim) wspomagającym podejmowanie decyzji na podstawie informacji zgromadzonych w bazie danych.

Istotną cechą systemów informatycznych jest możliwość ich przenoszenia pod inny system operacyjny i na inny sprzęt. Cecha ta umożliwia łatwe rozbudowywanie potencjału systemu w drodze wymiany sprzętu na silniejszy oraz proste (i w rezultacie oszczędzające koszty) powielanie systemu w innych środowiskach sprzętowych. Jednocześnie cecha ta umożliwia ekonomiczniejsze rozwiązanie procesu realizacji systemu. W trakcie projektowania, implementacji i testowania systemu można wykorzystywać mniejszą (czytaj: tańszą) konfigurację sprzętu, a dopiero po otrzymaniu sprawnego systemu informatycznego instaluje się konfigurację docelową.

Podane cechy strukturalne i funkcjonalne systemów informatycznych można uzyskać bez kłopotów, stosując koncepcję „systemu otwartego" (por. rozdz. 10), zalecanego przez rząd USA i agencje EWG.

Zakres i warunki informatyzacji administracji

Informatyzować można wszystko, pytanie tylko: po co, jakim kosztem i jakie to da efekty? Obecnie krystalizują się trzy kierunki informatyzowania administracji:
– **biurotyka,** oznaczająca informatyzowanie stanowiska pracy urzędnika przez wyposażenie go w mikrokomputer biurowy (z reguły osobisty typu IBM PC) i odpowiednie oprogramowanie standardowe (procesor tekstów, arkusz kalkulacyjny, prostą bazę danych, pocztę elektroniczną itp.) funkcjonujące w języku polskim;
– **reinformatyzacja,** polegająca na częściowej lub całkowitej modyfikacji istniejących systemów informatycznych, spowodowanej reorganizacją instytucji, stosowanych klasyfikacji czy koniecznością wymiany sprzętu;
– **informatyzacja,** polegająca na projektowaniu całkowicie nowych systemów informatycznych.

Obecnie wprowadzanie biurotyki wydaje się najbardziej zaawansowane, chociaż z braku odpowiedniego przeszkolenia, używania przypadkowego, nie licencjonowanego oprogramowania często nie spełnia to oczekiwań i w rezultacie zniechęca. Reinformatyzacja dotyczy praktycznie wszystkich istniejących systemów informatycznych. W zasadzie przy konieczności wycofania wszystkich maszyn serii Riad, wątpliwej wartości maszyn „second hand" typu IBM 43xx oraz przy przestarzałej strukturze wewnętrznej tych systemów (hierarchiczne lub sieciowe bazy danych, zapis w asemblerze lub Cobolu) warto rozważyć celowość tych działań. Tańsze będzie konstruowanie nowych systemów niż modyfikacja starych, tym bardziej że wielu ich autorów nie ma już w kraju. Plusem starych systemów jest jedynie dość dobra specyfikacja ich funkcji i wymagań użytkowych; jest także określona ich złożoność i stopień wykorzystywania.

Zapotrzebowanie na nowe systemy informatyczne wynika z nowych struktur organizacyjnych państwa. Obecnie takimi potrzebami są:
– system podatkowy, niezbędny do sprawnego pobierania podatku dochodowego i od wartości dodanej;
– systemy administracji lokalnej ewidencji ludności, pojazdów, bezrobotnych, ewidencji gruntów – upraszczające sprawne funkcjonowanie samorządów, a jednocześnie umożliwiające agregację tych informacji w skali państwa;
– system obsługi celnej – obecnie dopiero realizowany, poczynając od zakupu mikrokomputerów (!).

Oprócz tego istnieją pomysły na wiele nowych systemów, takich jak informacji sejmowej, analiz gospodarczych, analiz społecznych – praktycznie każda jednostka państwa chce mieć własny system. Pytanie tylko: czy państwo tych wszystkich systemów potrzebuje i czy je na to stać?

Odpowiedź na postawione pytanie jest prosta: teraz i w przyszłości państwo nie będzie dysponować wystarczającymi środkami na realizację wszystkich systemów. Należy więc wybierać, a przede wszystkim koncentrować środki na najważniejszych celach, nie marnotrawiąc ich jednocześnie na próby tworzenia tzw. faktów dokonanych.

Po decyzji o budowie nowego systemu należy zagwarantować środki na cały czas realizacji projektu, określając, że np. główne zakupy sprzętu mogą nastąpić tuż przed osiągnięciem wersji eksploatacyjnej systemu informatycznego. Złym nawykiem jest bowiem przedwczesne gromadzenie jeszcze niepotrzebnego sprzętu, ale nawyk (tak częsty w poprzednich czasach) można wyrugować tylko w drodze rzeczywistej gwarancji kwot wydzielonych na dany cel. Jednocześnie należy wymagać, aby projekt był realizowany zgodnie z zasadami organizacji przedsięwzięć informatyzacyjnych (por. rozdz. 4).

Odpowiedzialność administracji za informatyzację

Informatyzując administrację państwo podejmuje się jednocześnie przestrzegania podstawowych zasad ochrony gromadzonych informacji (por. rozdz. 11).

Na pierwszym miejscu stawiamy tutaj ochronę tajemnicy państwowej w sferze obronności i gospodarczej. Obecnie – przy szeroko otwartych granicach i wielu kontaktach międzynarodowych oraz dostępie zachodnich konsultantów do wielu danych o polskiej gospodarce – jest konieczne określenie minimum pakietu informacji, których naszym partnerom ujawniać nie wolno. Dodatkowym utrudnieniem spełnienia tych wymagań żywotnych dla polskich interesów jest fakt, że nie ma dziś społecznej świadomości potrzeby takiego postępowania. Tym bardziej więc ochrona ta musi być wmontowana w system informatyczny, i to przez zespół polskich specjalistów.

Drugim rodzajem ochrony jest ochrona dóbr osobistych osób, o których informacje są gromadzone w systemie. Ochrona ta powinna polegać na udostępnianiu danemu urzędnikowi tylko informacji mu niezbędnych, na blokadzie dostępu do informacji personalnych dla osób postronnych oraz określaniu dopuszczalnego zakresu gromadzenia informacji personalnych w ramach jednego systemu. Dla uniknięcia uzyskiwania informacji personalnych ponad niezbędne potrzeby (np. nie wolno łączyć systemu ewidencji medycznej z systemem ewidencji ludności) trzeba wypracować zakres i kontrolę przestrzegania łączenia wielu zbiorów informacji. Dodatkowym elementem ochrony dóbr osobistych jest określenie zasady identyfikacji osób. Dotychczas używany identyfikator (zapisany w dowodzie osobistym) narusza dobra osobiste, gdyż ujawnia datę urodzenia. Problem ten jest obecnie analizowany przy realizacji systemu POLTAX.

Kolejną powinnością państwa jest zagwarantowanie rzetelności funkcjonowania systemu, polegającej na 100% pewności w prawidłowości działania zarówno na rzecz każdego obywatela, jak i administracji. Na przykład musi być gwarantowana poprawność naliczania podatków, rzetelność informacji personalnych, informowania o stanie prawnym, księgowania wpływów i wydatków itp. W opisanych przypadkach państwo odpowiada nie tylko za swoje systemy, lecz także za wszystkie inne systemy działające zarówno w sferze budżetowej, jak i prywatnej.

Zagrożenia

Pozostawienie dotychczasowych zasad komputeryzacji administracji państwowej prowadzi do dalszego marnotrwawienia środków, coraz większego bałaganu w nie przystających do siebie systemach informatycznych oraz do zamulenia instytucji często przypadkowym sprzętem mikrokomputerowym. Jednocześnie dopuszczanie do eksploatacji słabych, projektowanych bezładnie i nie atestowanych systemów doprowadzi do powstawania strat finansowych oraz braku akceptacji zarówno ze strony urzędników, jak i społeczeństwa. Na przykład źle funkcjonujący informatyczny system podatkowy może wzmocnić brak akceptacji społecznej powszechnego podatku dochodowego. Groźne są też możliwości powstawania nadużyć w wykorzystywaniu informacji gromadzonych przez te, słabo chronione przed nieupoważnionym dostępem, systemy. Obecna działalność Biura do spraw Informatyki URM ma na celu zapobieżenie tym zagrożeniom, ale bez silnego wsparcia legislacyjnego i finansowego będzie zbyt słaba w opanowaniu istniejącego bałaganu.

Poważnym problemem jest brak dobrej i bardzo dobrej profesjonalnej kadry informatycznej (z niewielkimi wyjątkami) pracującej w administracji państwowej. Spowoduje to podjęcie wielu nietrafnych decyzji oraz uniemożliwi precyzyjne wyspecyfikowanie potrzeb, a następnie efektywne wdrożenie gotowych systemów (por. rozdz. 3). Jednocześnie brak tej kadry uniemożliwia podejmowanie fachowych negocjacji z partnerami zagranicznymi i krajowymi, ewentualnymi wykonawcami systemów i dostawcami oprogramowania oraz sprzętu.

Zalecenia

1. Podstawą opracowania założeń informatyzacji administracji państwowej powinna być strategia gospodarcza rządu. Niestety takiej strategii nie ma, a więc z natury rzeczy informatyzacja administracji musi odbywać się po omacku, pociągając za sobą konieczność bardzo precyzyjnego postępowania, aby skutki ewentualnych złych decyzji nie były w przyszłości zbyt dotkliwe.

2. Z zestawu systemów informatycznych dla administracji należy wydzielić grupę systemów, które muszą być projektowane i realizowane centralnie, a następnie po atestacji wdrażane w jednostkach lokalnych. W grupie tej powinny się znaleźć systemy ewidencji i ruchu ludności, podatkowe, obsługi socjalnej, ewidencji transportu, statystyki i analiz gospodarczych. Konieczne jest zwłaszcza sprecyzowanie powiązań systemów administracji samorządowej z systemami resortowymi. Pozostałe systemy mogą być zamawiane i opracowywane lokalnie, przy czym koniecznie muszą być atestowane przez agencję niezależną od zainteresowanych.

3. W projektowaniu, modyfikacji i wdrażaniu systemów informatycznych oraz ich wyposażenia sprzętowego i programowego jest konieczne zastosowanie wszystkich zaleceń dotyczących organizacji przedsięwzięć informatyzacyjnych (por. rozdz. 4). Jest także niezbędne opracowanie zasad negocjacji i podpisywania kontraktów z firmami dostarczającymi sprzęt i oprogramowanie. Szczególnie istotne jest opracowanie zaleceń dotyczących zasad odbioru, gwarancji, dostaw kolejnych wersji oraz miejsca rozstrzygania sporów (np. w Europejskiej Komisji Arbitrażowej zamiast przez sąd w Kalifornii).

4. Należy unikać równoczesnej realizacji tych samych systemów. Jedynie może być dopuszczalne dublowanie prac (przy jasnych kryteriach oceny) w celu szybkiego osiągnięcia najlepszego wyniku.

5. Należy dążyć do unifikacji systemów, oprogramowania i sprzętu komputerowego w drodze przyjęcia koncepcji systemów otwartych, o dużym stopniu elastyczności. Warto też dokonać specjalistycznej analizy ofert oprogramowania narzędziowego i sprzętu komputerowego, wybierając gamę zestawów użytecznych w administracji. Dla innych systemów, mających w przyszłości współpracować z systemami administracji państwowej należy szybko zainicjować, a następnie kontrolować procedury ustalania norm związanych bezpośrednio z informatyką oraz norm uzupełniających (por. zalecenia w rozdz. 10).

6. Należy podjąć decyzje administracyjne o całkowitym wyeliminowaniu z użytkowania w administracji nie licencjonowanego oprogramowania (zanim nawet powstanie odpowiednia ustawy o ochronie oprogramowania).

7. Należy zainicjować prace badawcze, a następnie opracować normy atestacji i walidacji systemów informatycznych, oprogramowania i sprzętu instalowanego na potrzeby administracji.

8. Jest też konieczne zainicjowanie prac nad ustawą o ochronie dóbr osobistych osób, o których informacje znajdują się w zbiorach informatycznych (por. rozdz. 11). Jednocześnie należy podjąć działania nad zorganizowaniem Komitetu Ochrony Informacji (nazwa robocza), którego zadaniem powinno być m.in.:
- nadzorowanie prac nad ustawą o ochronie informacji ze względu na dobro osobiste oraz interes państwa;
- weryfikowanie wdrażanych systemów informatycznych, gromadzących dane osobowe oraz informacje poufne, pod kątem ich ochrony przed nieuprawnionymi użytkownikami;
- kontrolowanie funkcjonujących systemów ze względu na sposób wykorzystywania zawartych w nich informacji.

9. Należy zainicjować prace nad ustawowym określeniem zakresu ważności prawnej dokumentów bezpośrednio uzyskiwanych z systemów informatycznych. Z tymi pracami wiążą się ustalenia dotyczące zasad tworzenia i przechowywania kopii i archiwizacji zbiorów oraz metod usuwania informacji nieaktualnych.

10. Warto jest zorganizować (rządowe lub we współpracy z uczelnią bądź PTI) centrum szkolenia pracowników administracji państwowej (centralnej i lokalnej) w zakresie podstaw użytkowania informatyki oraz pragmatyki jej stosowania. Centrum to może też doszkalać państwowych pracowników informatyki w nowych technologiach projektowania, implementacji i eksploatacji systemów oraz sprzętu informatycznego. Pracownikami centrum powinna być dobrze wyszkolona kadra z doświadczeniem dydaktycznym.

Rozdział 7

Sieci komputerowe

Wprowadzenie

Charakterystyczną cechą większości systemów informatycznych eksploatowanych i projektowanych obecnie w krajach wysoko uprzemysłowionych jest ich rozproszenie. Dotyczy ono punktów dostępu do systemu, zasobów informacyjnych systemu i zdolności ich przetwarzania. Cecha rozproszenia jest wynikiem wymagań zastosowań, które z natury rzeczy obejmują swym zasięgiem pewien obszar, np. przedsiębiorstwo, miasto, województwo, kraj lub świat. Przykładami dziedzin wymagających rozproszonych systemów informatycznych są: bankowość, państwowe służby finansowe, służby celne i graniczne, nauka, handel zagraniczny, policja, wojsko, transport itp.

Podstawą budowy rozproszonych systemów informatycznych są sieci komputerowe. W zależności od wielkości obejmowanego obszaru, wyróżniamy sieci komputrowe lokalne, metropolitalne i rozległe. Skala przedsięwzięcia, jakim jest budowa sieci, rośnie gwałtownie wraz z wielkością obejmowanego przez nią obszaru. Głównie z tego względu w chwili obecnej w Polsce szerzej są stosowane tylko sieci lokalne o szybkościach standardowych 1 do 10 Mbitów/s, łączące mikrokomputery. Dla poprawnego funkcjonowania wymienionych powyżej dziedzin jest natomiast konieczna dostępność sieci rozległych i metropolitalnych. Zwłaszcza dostępność sieci rozległych jest palącą potrzebą. Potrzeba ta jest rozumiana w niektórych środowiskach w Polsce, o czym świadczą podjęte próby budowy takich sieci (np. sieć POLPAK, międzynarodowa sieć akademicka EARN i sieć bankowa TELBANK).

W krajach wysoko uprzemysłowionych w odniesieniu do sieci rozległych mamy do czynienia z dwiema równolegle rozwijanymi technologiami: technologią sieci z komutacją pakietów i technologią sieci cyfrowych z integracją usług.

Sieci z komutacją pakietów

Sieci z komutacją pakietów, w których zastosowano protokół X.25 i które cechują się szybkością transmisji 9.6 do 64 Kbitów/s, są obecnie najbardziej rozpowszechnionym typem sieci rozległych. Gdy sieci te mają charakter publiczny, wówczas świadczone przez nie usługi komunikacyjne są dostępne dla dowolnych klientów, gdy natomiast mają charakter prywatny, wówczas świadczone przez nie usługi są dostępne tylko dla właściciela sieci i jego partnerów. Właścicielami sieci prywatnych są banki, policja, wojsko, uniwersytety, służby celne itp. Jednym z głównych motywów budowy tych sieci jest zapewnienie poufności przesyłanych danych. Sieci z komutacją pakietów są zalecane przez EWG do komunikacji między jej instytucjami a administracjami państw członkowskich.

Sieci cyfrowe z integracją usług

Sieci cyfrowe z integracją usług (ISDN – *Integrated Services Digital Network*) są wynikiem dążenia do integracji sieci komputerowych z sieciami telekomunikacyjnymi. Sieci te umożliwiają przesyłanie fonii, danych komputerowych, tekstu (np. teleks, teletekst, poczta elektroniczna) i obrazów (np. telefaks). Sieci te są stopniowo wprowadzane do praktyki wraz z postępującą cyfronizacją telekomunikacji. Wprowadzanie to odbywa się szybciej w krajach o przestarzałej sieci telekomunikacyjnej (np. we Francji i w Belgii), a wolniej w krajach o dobrze zorganizowanej sieci telekomunikacyjnej i dobrej jakości usług telekomunikacyjnych (np. w USA).

Sieci typu ISDN wprowadzają nową jakość w dostępie do sieci komputerowych – są one natychmiast dostępne tam, gdzie dociera cyfrowa sieć telefoniczna. Natomiast pod względem szybkości transmisji są one zaledwie porównywalne z sieciami pakietowymi. Ponieważ w wielu zastosowaniach praktycznych potrzeba znacznie większych szybkości transmisji, jest intensywnie rozwijana technologia tzw. szerokopasmowych sieci cyfrowych z integracją usług (B-ISDN – *Broadband Intergrated Services Digital Network*). Sieci te cechują się szybkościami od 100 do 600 Mbitów/s, a ich techniczne możliwości sięgają gigabitów na sekundę. W dziedzinie informatyki sieci B-ISDN pozwalają na uzyskanie na dużych odległościach lepszej jakości usług niż w standardowych sieciach lokalnych. W dziedzinie telekomunikacji sieci umożliwiają przesyłanie ruchomych obrazów, a zatem realizację usług dla telewizji kablowej o dużej rozdzielczości, videotelefonu itp. Ponieważ budowa powszechnych sieci B-ISDN jest sprawą przyszłości, a znaczna część zastosowań informatyki wymaga już teraz dużych szybkości transmisji, następuje gwałtowny rozwój sieci o dużych przepustowościach, świadczących wyłącznie usługi komputerowe. Mają one charakter sieci rozległych – nazywanych wówczas sieciami kręgosłupowymi (*backbone network*) lub sieci metropolitalnych, np. wykonanych w technologii DQDB (*Distributed-Queue Dual Bus*), bądź też lokalnych, np. wykonanych w technologii FDDI (*Fiber Distributed Data Interface*). Niezależnie od wielkości obszaru, sieci te stosuje się przede wszystkim do łączenia sieci lokanych i superkomputerów.

Podsumowanie

Wobec zarysowanego powyżej stanu i tendencji rozwojowych sieci komputerowych w krajach wysoko uprzemysłowionych, strategicznym celem rządu Rzeczypospolitej Polskiej powinno być zapewnienie instytucjom państwowym i prywatnym powszechnego dostępu do krajowych sieci komputerowych. Niezrealizowanie tego celu spowoduje blokadę informacyjną Polski i uniemożliwi stosowanie nowoczesnych technik zarządzania, zwłaszcza w sferze finansowej, co w konsekwencji uniemożliwi integrację Polski z Europą zachodnią i współpracę z innymi krajami wysoko uprzemysłowionymi.

Szczegółowe cele do osiągnięcia są następujące.

1. Należy zbudować, wdrożyć i rozpowszechnić usługi sieci publicznej z komutacją pakietów X.25 i stymulować rozwój takich sieci prywatnych. Budowana od długiego czasu sieć publiczna POLPAK w swych obecnych założeniach nie spełnia wymagań powszechności dostępu. Jest konieczne rozszerzenie jej koncepcji i zintensyfikowanie budowy. Warto z góry zwrócić uwagę na trudności, jakie pojawią się przy budowie sieci z komutacją pakietów – zarówno publicznej, jak i prywatnych. Dużą trudnością będzie dostęp do węzłów sieci, zwłaszcza w dużych ośrodkach miejskich, co wynika ze słabości i złej struktury sieci telekomunikacyjnej w tych ośrodkach. Inną trudnością będzie zapewnienie szybkich połączeń między centrami zawierającymi duże komputery i sieciami lokalnymi. Właściwym rozwiązaniem tych trudności byłaby budowa szybkich sieci metropolitalnych i międzymiastowych sieci kręgosłupowych przez różnych inwestorów. Zdając sobie jednak sprawę z kosztowności takiego przedsięwzięcia, proponujemy:
   - prowadzenie promocyjnej polityki stymulującej powstawanie prywatnych, miejskich i międzymiastowych, sieci kręgosłupowych stosujących technologię sieci lokalnych i technologię sieci cyfrowych (kanały 2 Mbity/s);
   - zrealizowanie pilotowego rozwiązania takiej sieci dla wybranego działu administracji państwowej;
   - uruchomienie wdrożeniowych programów badawczych dotyczących sieci metropolitalnych i kręgosłupowych. Byłoby celowe, aby programy te były realizowane wspólnie z podobnymi programami w ramach współpracy EWG (np. program BERKOM).
2. Należy zintensyfikować proces przyjmowania jako polskich norm odpowiednich standardów dotyczących sieci komputerowych. Standardy te obejmują normy podstawowe, dotyczące specyfikacji usług i protokołów sieci, i normy funkcjonalne, nazywane profilami. Za normy podstawowe należy przyjąć odpowiednie standardy ISO i CCITT związane z tzw. modelem odniesienia dla współpracy systemów otwartych (OSI RM). Obowiązujące w Polsce profile powinny być wybrane spośród profili zalecanych dla EWG przez CEN/CENELEC (Europejski Komitet Normalizacyjny / Europejski Komitet Normalizacji Elektronicznej). Profile dla CEN/CENELEC są opracowywane przez zespoły ekspertów EWOS (European Workshop on Open Systems) i ETSI (European Telecommunications Standards Institute). W związku z tym Polska powinna stać się aktywnym, pełnoprawnym uczestnikiem odpowiednich komitetów technicznych i grup roboczych ISO, CCITT, EWOS i ETSI oraz

uzyskać prawo czynnego udziału w projekcie COSINE (Co-operation for Open Systems Interconnection Networking in Europe, EUREKA Project) prowadzonego przez RARE (Réseau Associés pour la Recherche Européene).

3. W związku ze standaryzacją sieci jest konieczne wprowadzenie obowiązku homologacji urządzeń i oprogramowania sieciowego, potwierdzanie ich zgodności z przyjętymi standardami i profilami, a także spełnianie przez nie wymagań jakościowych. Urządzenia i oprogramowanie sieciowe powinny być także badane pod kątem ich przyszłej zgodności lub możliwości integrowania z sieciami typu ISDN i B-ISDN. W odniesieniu do tej tematyki Polska powinna włączyć się do prac projektu CTS (Conformance Testing Services) finansowanego przez EWG i poszczególne PTT (poczty).

4. Proponuje się powołanie Państwowej Agencji Sieci Komputerowych, na wzór Państwowej Agencji Radiokomunikacyjnej i Państwowej Inspekcji Telekomunikacyjnej. W zakresie działania agencji powinien znaleźć się całokształt spraw związanych ze standaryzacją sieci, homologacją urządzeń i oprogramowania sieciowego oraz kontroli działania sieci, ochrony sieci itp.

5. Integracja sieci polskich z sieciami międzynarodowymi wymaga prowadzenia właściwej polityki na arenie międzynarodowej. Polska, z jednej strony, musi zadbać o ochronę swoich danych, a z drugiej – czynić starania o maksymalne rozluźnienie ograniczeń w dostępie do danych i usług oferowanych w sieciach międzynarodowych. Obecnie ograniczenia te są ciągle znaczne.

6. Ochrona danych w sieciach polskich wymaga własnych, oryginalnych rozwiązań. Wynika to z faktu, że zaawansowanych systemów ochrony, z jednej strony, za granicą kupić nie można, a z drugiej strony, za granicą kupować nie warto, ze względu na ich poufny charakter. Prace nad ochroną sieci komputerowych powinny być prowadzone pod nadzorem rządu i przezeń finansowane.

7. Należy określić prawną odpowiedzialność użytkowników i operatorów sieci komputerowych za włamania do systemów, wprowadzanie wirusów i inne przestępstwa informatyczne (por. rozdz. 11).

8. Rozwój sieci komputerowych, który ma kluczowe znaczenie dla rozwoju kraju, będzie zależał w decydującej mierze od kadr. Kadry te należy wykształcić w szkołach wyższych, co wymaga od rządu preferencji dla kierunków „informatyka" i „telekomunikacja". Preferencje te nie mogą być werbalne, lecz muszą być materialne (uposażenia, wyposażenia laboratoriów, lokale itp.).

Rozdział 8

Firmy informatyczne

Rodzaje firm informatycznych

Wyróżniamy pięć kategorii firm zajmujących się informatyką:

I. Państwowe fabryki i instytuty przemysłu komputerowego.

II. Firmy prywatne (rzemieślnicze, spółki, fundacje) w znacznym stopniu zajmujące się produkcją informatyczną (najczęściej oprogramowaniem).

III. Firmy zajmujące się handlem sprzętem komputerowym i oprogramowaniem.

IV. Firmy zajmujące się szkoleniami i wydawnictwami o tematyce informatycznej.

V. Zagraniczne firmy informatyczne mające swoje przedstawicielstwa lub oddziały w Polsce.

Przemysł komputerowy

Polski przemysł komputerowy praktycznie już nie istnieje. Obowiązujący do niedawna podział zadań w RWPG wyznaczył Polsce produkcję przestarzałych komputerów R-32 i R-34, minikomputerów SM-4 oraz starych technologicznie urządzeń peryferyjnych. W połowie lat osiemdziesiątych Zakłady ELWRO rozpoczęły montaż z części mikrokomputerów typu IBM PC, mając równocześnie licencję na sprzedaż systemu MS-DOS (co jest chyba jedyną jasną stroną tego przedsięwzięcia). Obecnie fabryki przemysłu komputerowego zmniejszają produkcję już przestarzałego sprzętu, gdyż nie wytrzymują konkurencji ze sprzętem z Dalekiego Wschodu i ze względu na załamanie współpracy z ZSRR. Pojedyncze kontrakty nie są w stanie zapewnić wystarczających środków na dalsze inwestycje. Również reklamowane powstanie koncernu ELPOL jest jedynie pretekstem lobby elektronicznego do podtrzymania nieefektywnej struktury przemysłu komputerowego.

Podobnie zanikł przemysł produkcji elementów elektronicznych i układów scalonych o małej skali integracji. Na obecnym poziomie technologii polski przemysł półprzewodnikowy nie ma żadnych szans. Jego dobre czasy już minęły, i to chyba bezpowrotnie. Jedyną jeszcze szansą jest współpraca z partnerami zagranicznymi w celu montażu urządzeń komputerowych oraz produkowanie specjalistycznych urządzeń o dużym wkładzie myśli technicznej.

Również głęboki kryzys przeżywają instytuty badawcze związane z informatyką (Instytut Maszyn Matematycznych, Instytut Systemów Sterowania, Instytut Organizacji Przemysłu Maszynowego, Instytut Informacji Naukowo-Technicznej i Ekonomicznej i inne). Instytuty te miały ograniczone możliwości rozwoju, gdyż dla produkowanych komputerów jedynie przystosowywano oprogramowanie systemowe ze wzorców produkowanego sprzętu. Nie stymulowano również rozwoju zastosowań informatyki. Obecnie instytuty te, zbytnio rozbudowane administracyjnie, nie mogą być konkurencyjne dla prywatnych spółek. Szybko też tracą wykwalifikowaną kadrę.

Firmy prywatne

Obecnie w Polsce działa:

- kilkadziesiąt prywatnych firm zajmujących się produkcją oprogramowania, głównie aplikacji dla prostych baz danych;
- około stu prywatnych polskich firm komputerowych zajmujących się przeważnie pośrednictwem w sprzedaży sprzętu komputerowego z Dalekiego Wschodu oraz ostatnio również z Europy i USA.

Wszystkie te firmy znajdują się w słabej kondycji finansowej i muszą się ratować handlem, i to nawet innych produktów (faksów, telefonów itp.)). Brak możliwości kredytowych, załamanie się rynku wschodniego oraz trudna sytuacja gospodarcza powodują dużą niestabilność istnienia tych firm.

Równocześnie daje się zauważyć zjawisko odśrodkowego działania wewnętrznego i zewnętrznego w tych firmach. Brak jasnych zasad podziału praw autorów i inwestora (właściciela firmy) do realizowanych produktów powoduje rozbijanie zespołów. Pracownicy tych firm szybko też dążą do usamodzielnienia się, co powoduje dalsze zmniejszanie się siły inwestycyjnej tych firm. Co więcej, firmy te nie są skłonne do współpracy dla realizacji większych przedsięwzięć lub wspólnego poszukiwania partnerów zagranicznych.

Inwestycje w sferze informatyki są kapitałochłonne (wysokie koszty zakupu dobrego sprzętu i licencjonowanego, profesjonalnego oprogramowania), wymagają szybkiego odtwarzania (krótki okres amortyzacji), a okres ich zwrotu jest dość długi (wykonanie porządnego, niezbyt skomplikowanego programu wymaga co najmniej roku wytężonej pracy). Pomimo nawału ofert na sprzedaż sprzętu mikrokomputerowego oraz oprogra-

mowania narzędziowego i prostych aplikacji, sytuacja tego rynku nie rokuje samoistnego powstania silnych, samodzielnych firm mogących sprostać kompleksowemu zapotrzebowaniu na efektywną profesjonalną informatykę. Warto jednak pamiętać, że w wielu z tych firm znajdują się dobre zespoły programistów oraz nieźli menedżerowie.

Firmy szkoleniowe i wydawnicze

Zapotrzebowanie na podstawową wiedzę informatyczną jest ogromne. Żadne zachodnie przedstawicielstwo i żadna poważna polska firma nie zatrudnią nawet sekretarki bez znajomości obsługi faksu i procesora tekstów. Takiego szkolenia potrzebuje również wielu urzędników. Jednocześnie w interesie państwa powinno leżeć propagowanie w społeczeństwie zastosowań informatyki. Wzorem dla nas powinien być tutaj prezydent Bush, który swoim przykładem nauczenia się obsługi komputera zachęca społeczeństwo amerykańskie do poznania tej dziedziny, która będzie miała podstawowy wpływ na jego życie w najbliższych latach.

Dlatego też nieco inaczej należy traktować firmy zajmujące się tylko szkoleniami i wydawnictwami. Niestety, większość firm prywatnych poponujących kursy szkoleniowe dotyczące użytkowania mikrokomputrów nie dysponuje zespołami o odpowiednich umiejętnościach dydaktycznych. Wbrew ogólnemu mniemaniu nauczenie początkujących, często starszych wiekiem urzędników posługiwania się komputerem wymaga większych umiejętności niż uczenie podstaw informatyki na uczelni.

Firmy zagraniczne w Polsce

Obecnie w Polsce działa kilkanaście samodzielnych przedstawicielstw (lub spółek typu joint venture) informatycznych firm zagranicznych. Ich podstawowym zadaniem jest promocja produktów i maksymalizacja zysków ze sprzedaży. Część z nich ma ambitniejsze plany łączenia promocji produktów ze współpracą z polskimi firmami w tworzeniu polskich wersji i aplikacji. Tym planom należy jak najbardziej sprzyjać.

Inne firmy zagraniczne organizują swoje sieci sprzedaży za pośrednictwem polskich firm stających się ich dystrybutorami lub ajentami. W ten sposób stało się możliwe kupienie legalnych kopii wielu różnych (od dawna już w Polsce używanych) programów i systemów. Istnieją jednak inne aspekty takiego rozwiązania. Przede wszystkim słabość finansowa polskich firm i niechęć do współpracy zmniejszają ich szanse wynegocjowania dobrych warunków dystrybucyjnych, dyktowania warunków dostaw i zasad współpracy. Szefowie polskich firm (często absolwenci studiów informatycznych) mają zbyt małe doświadczenie w takich kontaktach z wytrawnymi menedżerami zagranicznymi. Dodatkowo brak ustawy o ochronie oprogramowania, piracki rynek i drogie kredyty utrudniają negocjacje, a potem efektywną działalność, gdzie dystrybutor najpierw musi zainwestować, a dopiero potem może liczyć na prowizje.

Zagrożenia

W dziedzinie informatyki nie jesteśmy już konkurencją dla kogokolwiek liczącego się na świecie. Co więcej, sami możemy zostać całkowicie zdominowani przez firmy zagraniczne, które zajmą newralgiczne pozycje w każdej dziedzinie działalności gospodarczej, administracyjnej i społecznej (politycznej również). Obecnie mamy jeszcze szansę włączenia do procesu unowocześniania krajowej informatyki polskich zespołów programistów, projektantów, menedżerów i techników. Pozbawienie się tej szansy spowoduje skutki praktycznie nieodwracalne w następnych latach. Szybkie wchłonięcie naszej gospodarki w obieg systemów informatycznych Europy i świata bez naszego udziału uniemożliwi w przyszłości tworzenie własnej infrastruktury informatycznej.

Zalecenia

1. Porównanie efektów rozwoju informatyki amerykańskiej (rząd USA finansuje zastosowania) z rozwojem informatyki zachodnioeuropejskiej (rządy EWG dofinansowują przemysł informatyczny) wyraźnie wskazuje na wyższość rozwiązania z finansowaniem zastosowań. Dlatego też nie wolno ratować polskiego przemysłu komputerowego dofinansowując go z budżetu. Pieniądze na rozwój informatyki należy przekazywać tym, którzy mają ją stosować.
2. Państwowe przedsiębiorstwa przemysłu komputerowego powinny podlegać zwykłej grze sił rynkowych i konkurować na równych prawach z innymi podmiotami gospodarczymi, krajowymi i zagranicznymi. Tylko taka strategia może, naszym zdaniem, wymusić poważne zmiany strukturalne w polskim przemyśle komputerowym w drodze powiązania go z rynkiem światowym. Pożądany krok w tym kierunku stanowi nowa ustawa o inwestycjach zagranicznych, przyznająca preferencje inwestorom wprowadzającym nowe technologie. Należy tylko sprecyzować pojęcie rzeczywistej nowej technologii.
3. Informatyka może być motorem rozwoju gospodarczego. Rząd musi stale podkreślać znaczenie informatyki dla przyszłego rozwoju kraju. Wyrazem takiego poparcia powinny być odpowiednie działania na forum międzynarodowym (por. rozdz. 2), wspieranie inicjatyw odtwarzania polskiego rynku zastosowań informatyki wykorzystując kapitał własny i zagraniczny oraz przeznaczanie znacznej części nakładów na naukę, na rozwój badań podstawowych związanych z samą informatyką i jej zastosowaniami. Na tej podstawie mogą być dopiero odtwarzane podstawy przemysłu komputerowego i informatycznego.
4. Jest konieczne jednakowe traktowanie wszystkich podmiotów w sferze działalności informatycznej. W przetargach i konkursach firmy państwowe, prywatne i zagraniczne powinny mieć identyczne prawa. Szczególnie jednak należy popierać oferty, które łączą w sobie dostęp do technologii zachodniej z wykorzystaniem polskiego potencjału intelektualnego.
5. Należy wspierać powstawanie polskich zespołów konsultacyjnych, przygotowujących firmę do informatyzacji, a następnie opracowujących zapytania ofertowe, prowadzących przetargi i konkursy oraz negocjujących kontrakty.

Rozdział 9

Stymulowanie rynku zastosowań

Konieczność skutecznego konkurowania na rynkach światowych szybko zmusi polskie przedsiębiorstwa i instytucje wszystkich sektorów do poważnych inwestycji w zastosowania informatyki. Tak więc, mimo że bezpośrednie inwestycje państwowe w tworzenie zastosowań w sektorze publicznym pochłoną duże środki finansowe, główny ciężar inwestycyjny będą ponosić podmioty pozarządowe. Takie inwestycje leżą niewątpliwie w interesie państwa jako jeden ze środków zmiany struktury gospodarki, przezwyciężania kryzysu gospodarczego i dostosowania do standardów informacyjnych EWG. Rząd powinien więc stosować odpowiednie mechanizmy stymulacji przedsięwzięć informatyzacyjnych.

Największe koszty pochłonie niewątpliwie informatyzacja banków i innych instytucji finansowych. Istnieją również inne dziedziny, których informatyzacja jest bliska i nieuchronna. Dotyczy to m.in. wszystkich przedsiębiorstw spedycyjnych i eksportowo-importowych, ze względu na nowe przepisy obiegu dokumentów celnych i listów przewozowych w EWG, jakie będą obowiązywać od 1 stycznia 1992 r. Równie szybko informatyzacji muszą poddać się działy finansowo-księgowe wszystkich przedsiębiorstw, w związku ze zmianami w systemie podatkowym (podatek od wartości dodanej, powszechny podatek dochodowy) i zmienionym podejściem do gopodarki finansowej (analiza przepływów pieniężnych, zmienne kursy walutowe itp.). Wolna konkurencja na rynkach światowych zmusi też przedsiębiorstwa produkcyjne do powszechnego stosowania informatycznych środków projektowania nowych wyrobów i zarządzania produkcją (CAD, CAM).

Najważniejszymi mechanizmami stymulacji przedsięwzięć informatyzacyjnych stosowanymi przez rząd powinny być:

1. Skrócenie okresu amortyzacji informatycznych środków trwałych. Obecnie te okresy wynoszą od 7 lat (dla większych mikrokomputerów, minikomputerów i dużych komputerów) do 5 lat (dla najmniejszych mikrokomputerów) i nie pozwalają na szybsze odtwarzanie sprzętu zużytego moralnie i technicznie. Uregulowania prawnego wymaga też szybka amortyzacja wartości niematerialnych i prawnych związanych z inwestycjami w oprogramowanie podstawowe i zastosowawcze. Jednolite skrócenie maksymalnego okresu amortyzacji sprzętu komputerowego i oprogramowania co najmniej o połowę pozwoli na szybszą spłatę poniesionych nakładów dzięki zatrzymaniu większego zysku w informatyzujących się przedsiębiorstwach.

2. Generalne i trwałe zniesienie ceł na sprzęt informatyczny, podzespoły stosowane w komputerach, oprogramowanie i materiały informatyczne. Obecnie obowiązujące stawki celne są niejednolite i w dodatku oparte na nieaktualnej klasyfikacji, która np. dyskietki komputerowe zalicza do urządzeń do zapisu obrazu i dźwięku oraz każe clić oprogramowanie o wartości wielokrotnie przekraczającej wartość nośników, na których jest zapisane, według stawek dla kaset magnetofonowych. Cel wysokich stawek celnych na niektóre środki informatyczne jest przy tym mocno nieprzekonujący, ponieważ nie mają one odpowiedników krajowych, a przynajmniej odpowiedników o dobrej jakości i dobrych walorach funkcjonalnych.

3. Podniesienie dolnej granicy wartości środków informatycznych, powyżej której są one traktowane jako środki trwałe i nie mogą być zaliczone w koszty w chwili zakupu. Obecnie obowiązująca granica 5 mln zł uniemożliwia szybkie wliczenie w koszty większości oprogramowania licencjonowanego, jakie jest sprzedawane na polskim rynku.

4. Wyodrębnienie z puli kredytów zagranicznych przeznaczonych na restrukturyzację gospodarki części przeznaczonej na inwestycje informatyzacyjne oraz częściowe gwarantowanie przez rząd takich kredytów na jednakowych warunkach dla wszystkich sektorów gospodarki. Takie działania obniżą cenę kredytu na przedsięwzięcia informatyzacyjne, poprawiając ich rachunek opłacalności.

5. Wprowadzenie ułatwień w przepisach prawnych dotyczących leasingu finansowego i operacyjnego w odniesieniu do sprzętu i oprogramowania komputerowego. W krajach wysoko rozwiniętych znaczna część środków informatyczych jest użytkowana właśnie na zasadzie leasingu.

Proponowane mechanizmy będą wpływać pozytywnie również na wyniki polskich firm informatycznych (por. rozdz. 8), prowadząc do ich wzmocnienia i możliwości przyszłej skutecznej konkurencji na rynkach międzynarodowych, gdzie z powodzeniem działają już firmy węgierskie. Należy pamiętać, że działalność firm informatycznych należy głównie do sektora usług, którego udział w produkcie narodowym brutto (GNP) znacznie odbiega od takiego udziału w krajach uprzemysłowionych, a który przy tym charakteryzuje się świetną relacją zysków do poniesionych nakładów materialnych. Wzmocnione w ten sposób firmy informatyczne uzyskają środki niezbędne do inwestycji w produkcję oprogramowania użytkowego, czyli gałąź przemysłu informatycznego, której udział w gospodarce światowej rośnie bardzo szybko.

Instytut Maszyn Matematycznych szkoli użytkowników PC

W prasie codziennej pojawia się coraz więcej ogłoszeń o kursach szkoleniowych dla użytkowników komputerów osobistych klasy IBM PC. Zaostrzająca się konkurencja w dziedzinie sprzedaży sprzętu i oprogramowania tej kategorii komputerów powoduje, że wiele firm rozszerza działalność na bardzo potrzebne (bo rozszerzające rynek zbytu), szkolenie użytkowników. Skuteczność tego szkolenia jest jednak bardzo zróżnicowana, głównie ze względu na znaczne trudności z pozyskaniem dobrych wykładowców. Powoduje to, że przy wyborze instytucji szkolącej istnieje bardzo duże ryzyko popełnienia błędu zresztą nie tylko w Polsce. Liczne artykuły w czasopismach zagranicznych alarmują o coraz częstszych przypadkach prowadzenia działalności szkoleniowej nie gwarantującej opanowania oferowanych w ogłoszeniach umiejętności.

Niewątpliwie najmniejsze ryzyko wiąże się ze szkoleniem oferowanym przez placówki naukowo-badawcze lub znanych producentów sprzętu komputerowego i oprogramowania. W Polsce systematyczną działalność szkoleniową dla użytkowników komputerów klasy IBM PC podjął w grudniu 1989 r. Insytut Maszyn Matematycznych, pionierska i najbardziej zasłużona dla rozwoju polskiej informatyki placówka badawczo-rozwojowa. Instytut prowadzi w sposób ciągły tego rodzaju kursy o bardzo zróżnicowanym poziomie i tematyce szkolenia. Do chwili obecnej na kursach tych przeszkolono ponad 1100 osób, głównie pracowników instytucji rządowych. Największy udział i powodzenie mają nadal dwutygodniowe kursy podstawowe, obejmujące 40 godzin lekcyjnych, z których większość stanowią zajęcia praktyczne. Organizowane są również 1- lub 2-tygodniowe kursy specjalistyczne dla użytkowników bardziej zaawansowanych. Zajęcia odbywają się w siedzibie Instytutu przy ul. Krzywickiego w Warszawie, rano lub po południu (do wyboru). Oferuje się możliwość organizowania szkolenia również w siedzibie zleceniodawcy, a także ukształtowania programu kursu zgodnie z indywidualnymi potrzebami zainteresowanych osób. Do końca pierwszego kwartału br. odbędą się jeszcze następujące kursy:

- kurs XXXVII: podstawowy (DOS, NC, Windows, Dbase III, Chiwriter WordPerfect), – od 17 do 28.02,
- kurs XXXVIII: wydawnictwo na biurku (NC, Word, Ventura Publisher) – od 2 do 13.03,
- kurs XXXIX: serwis PC – od 9 do 20.03
- kurs XL: podstawowy (DOS, NC Windows, dBase III, Chiwriter oraz WordPerfect lub Lotus 1.2.3.) – od 23.03 do 3.04.

Zainteresowani szczegółami tej działalności Instytutu mogą zwracać się po informacje do **Ośrodka Badawczo-Szkoleniowego Technik Komputerowych, ul. Krzywickiego 34, 02-078 Warszawa, tel. 29-91-64 lub 21-84-41 w. 500, teleks 813517, telefaks 29-92-70.**

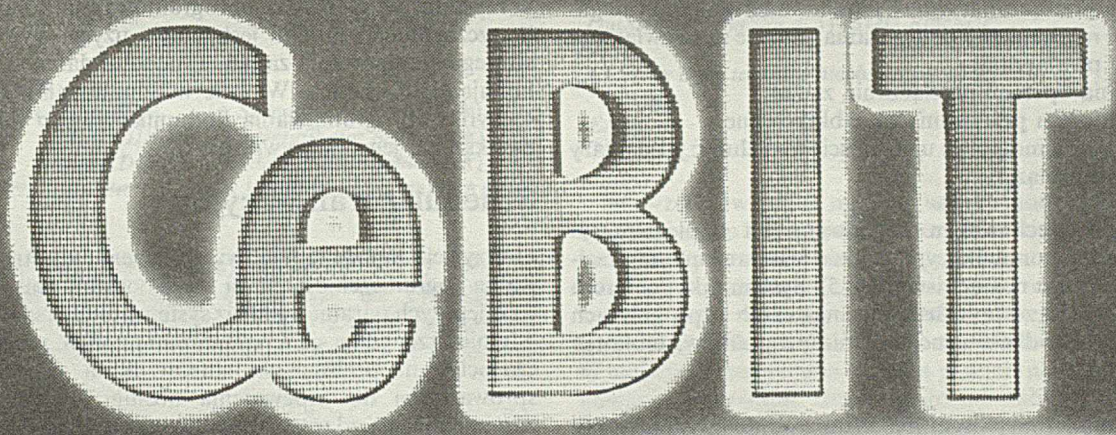
It's CeBIT Time

W Hanowerze planowana przekrojowa prezentacja inteligentnych komputerów

5.000 wystawców z 45 krajów przedstawi swoją ofertę w ramach całościowego przeglądu wszystkich obszarów techniki informacyjnej i komunikacyjnej. Oczekuje się także, że zaprezentowane zostaną zwykle nie eksponowane rozwiązania całościowe a także zaskakujące nowości.

Całą skalę otwierających się możliwości można objąć tylko uczestnicząc bezpośrednio w imprezie.

HANOWER
11.–18. MARZEC 1992



Światowe Centrum • Biuro • Informacja • Telekomunikacja

DEUTSCHE MESSE AG. HANNOVER/GERMANY

Bliższe informacje: POLEXPO ul. Lopuszanska 38, PL-02-232 Warszawa,
Tel.: (022) 46 04 01-9, Telex: Polex PL 813633, Fax: (022) 46 45 91

SO/710/91

Artykuł jest istotnie rozbudowaną wersją referatu wygłoszonego przez Autorów w ramach międzynarodowej konferencji „Computer Networks '91", która odbyła się we Wrocławiu w ubiegłym roku. W obecnej wersji, obok wyników badań, podano wszystkie szczegóły implementacji proponowanego schedulera w takiej formie, aby Czytelnik mógł zbudować podobny system dla własnych potrzeb.

ZBIGNIEW FRYŹLEWICZ
Centrum Obliczeniowe
Politechnika Wrocławska

JĘDRZEJ UŁASIEWICZ
Instytut Cybernetyki Technicznej
Politechnika Wrocławska

Scheduler transakcyjny dla zastosowań komunikacyjnych

Implementacja typowego protokołu komunikacyjnego o warstwowej budowie wymaga wyboru właściwego środowiska programistycznego ułatwiającego takie przedsięwzięcie. Oprogramowanie komunikacyjne jest zwykle konstruowane jako zbiór procesów asynchronicznych. Procesy są uaktywniane zarówno przerwaniami pochodzącymi od urządzeń zewnętrznych, jak i przez inne procesy. Generalnie, środowisko, w którym jest implementowane oprogramowanie komunikacyjne, powinno spełniać następujące wymagania:

- realizacja wielozadaniowości,
- komunikacja i synchronizacja międzyzadaniowa,
- obsługa przerwań,
- odmierzanie czasu.

Ważnym wymaganiem jest i to, by system mógł obsługiwać intensywny strumień przerwań w ograniczonym od góry czasie. Przekroczenie takiego limitu czasu prowadzi zwykle do utraty danych. Oprogramowanie komunikacyjne musi więc spełniać wymagania typowe dla systemu czasu rzeczywistego. Duża częstotliwość przełączania zadań narzuca szczególne wymagania na jakość mechanizmu przełączania. Istotne są tu zwłaszcza dwa aspekty przełączania:

- czas trwania operacji przełączania zadań,
- okres, w którym przerwania są zablokowane.

Powyższe kryteria mogą być ujęte ilościowo i służyć jako miary stosowanych rozwiązań.

Dość rozpowszechnionym podejściem jest implementacja oprogramowania komunikacyjnego z zastosowaniem uniwersalnego systemu czasu rzeczywistego [4, 5]. System taki dostarcza zwykle obszernego zestawu usług wymaganych w programach współbieżnych, środków uruchamiania programów, bibliotek i innych narzędzi.

Podejściem alternatywnym jest zastosowanie jako narzędzie języków, a raczej systemów programowania, takich jak Modula-2 lub Ada, zawierających w sobie niezbędne konstrukcje procesowe czy zadaniowe. W tym przypadku współbieżność jest dostarczana przez język programowania, a nie przez system operacyjny. Stąd programy współbieżne mogą być tworzone w środowisku jednozadaniowym. Typowym przykładem jest Modula-2, która umożliwia tworzenie programów współbieżnych w środowisku MS-DOS-a.

Oba te podejścia mają jednak jedną wspólną cechę. Podstawą działania jest tutaj taki lub inny scheduler przełączający procesy. Przełączenie procesów polega typowo na zachowaniu kontekstu procesu bieżącego, przełączeniu stosów i odtworzeniu kontekstu procesu wznawianego.

Niezależnie od typu użytego schedulera można łatwo zauważyć, że każdy z nich wprowadza dość znaczne narzuty, związane nieodłącznie z samym mechanizmem przełączania procesów. Zmniejsza to potencjalną przepustowość całego systemu przez to, że czas procesora zamiast na przetwarzanie komunikatów protokołu (-ów) jest zużywany przez procedury schedulera. Co gorsze, w dużym procencie tego czasu muszą być zablokowane wszystkie przerwania w systemie, by zapewnić integralność struktur danych schedulera. Czas ten – w zależności od zastosowanego procesora – może być aż rzędu pojedynczych milisekund. Przez tak długi okres system nie reaguje na żadne zdarzenia. W rezultacie mogą być gubione pojedyncze znaki (bloki) informacji przesyłane kanałami fizycznymi.

Można tutaj zauważyć, że pewne właściwości uniwersalnego systemu wielozadaniowego są nadmiarowe i pozostają niewykorzystane w typowym oprogramowaniu komunikacyjnym. W szczególności rzadko jest celowe użycie wywłaszczeniowej strategii szeregowania zadań. Typowe zadania są krótkie i wykonują się do końca. Właśnie w dopasowaniu schedulera do specyfiki oprogramowania komunikacyjnego leży możliwość zwiększenia jego efektywności.

Scheduler transakcyjny

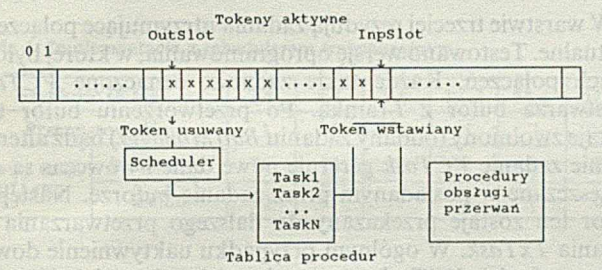
Przyjęcie założenia o niewywłaszczeniowej strategii szeregowania i ograniczenie zestawu usług pozwala zbudować scheduler o lepszych parametrach niż systemy uniwersalne. W sytuacji, gdy nie przewiduje się wywłaszczania zadań, nie jest potrzebne czasochłonne przełączanie stosów. Rola systemu może być sprowadzona do zapewnienia obsługi zdarzeń, a więc do zarejestrowania zdarzenia, a następnie wykonania, niekoniecznie natychmiast, procedury jego obsługi. W trakcie wykonywania procedury obsługi zdarzenia mogą nastąpić zgłoszenia nowych zdarzeń, na przykład przez procedury obsługi przerwań. System musi więc dostarczać mechanizm kolejkowania zdarzeń.

Prezentowane tutaj rozwiązanie nosi nazwę schedulera transakcyjnego [3]. W metodzie tej – pozbawionej z natury mechanizmu „czekaj na zdarzenie" – dopiero gotowe dane generowane przez dowolny proces wyzwalają przetwarzanie w innym procesie. Każdy proces, mający strukturę procedury, otrzymuje procesor na tak długo, jak tego potrzebuje. Przetwarza w tym

czasie dane, które spowodowały jego uaktywnienie, i (lub) wytwarza dane z przeznaczeniem dla innych procesów. Jeżeli proces taki jest przerywany, to obsługa takiego przerwania odbywa się na pojedynczym, wspólnym dla wszystkich stosie. Redukowany jest tym samym do minimum czas „nieaktywności" systemu. Powrót z przerwania następuje zawsze do przerwanego procesu. Ponieważ każdy proces ma formę procedury, nie mogą więc następować wzajemne przerwania procesów.

Implementacja schedulera transakcyjnego

Do implementacji zastosowano język TopSpeed Modula-2 [7]. Rozwiązanie jest oparte na metodzie przekazywania jednostek danych, zwanych tokenami, za pośrednictwem bufora cyklicznego (rys. 1). Zgłoszenie zdarzenia do obsługi polega na umieszczeniu w buforze odpowiedniego tokenu. Zgłoszenie to może być wykonane przez bieżące zadanie lub procedurę obsługi przerwania. Token, zdefiniowany poniżej, jest rekordem zawierającym identyfikatory zadania wysyłającego i odbierającego token, wskaźnik na pole danych wraz z ewentualną specyfikacją operacji do wykonania na tych danych. Operacje te zależą od specyfiki rozwiązywanego problemu i w definicji rekordu są podane w typie wyliczeniowym *ComType*. W zastosowaniach komunikacyjnych pole danych jest zwykle buforem mieszczącym jednostkę protokołu.



Rys. 1. Bufor cykliczny tokenów

```
TYPE
Token = RECORD
        Command: ComType   ; (★ Specyfikacja operacji ★)
        From     : CARDINAL; (★ Identyfikator nadawcy ★)
        To       : CARDINAL; (★ Identyfikator odbiorcy ★)
        DataPtr  : BufPtr   ; (★ Wskaźnik pola danych ★)
        END;
```

W systemie jest utrzymywany bufor cykliczny tokenów odpowiadających zgłoszonym, ale jeszcze nie obsłużonym, zdarzeniom tzw. tokenów aktywnych. Cykliczny bufor tokenów (*Ring*) oraz dwa związane z nim indeksy są zdefiniowane następująco:

```
VAR  Ring    : ARRAY [0..RingSize] OF Token;
     InpSlot : CARDINAL;
     OutSlot : CARDINAL;
```

Zmienna *InpSlot* wskazuje pozycję w buforze, w której powinien być umieszczony token odpowiadający nowemu zdarzeniu. Zmienna *OutSlot* określa pozycję tokenu odpowiadającego zdarzeniu, które będzie obsłużone w następnym cyklu schedulera. Zgłoszenie nowego zdarzenia następuje przez wykonanie systemowej procedury *Insert*(*Tkn*: *Token*). Umieszcza ona token *Tkn* w pozycji *InpSlot* bufora. Kod tej procedury jest następujący:

```
PROCEDURE Insert(Tkn: Token);
BEGIN
  DI;(★ Disable Interrupts ★)
  Ring[InpSlot] := Tkn;
  InpSlot := (InpSlot + 1) MOD RingSize;
  IF InpSlot = OutSlot THEN FatalError('Ring overflow') END;
  EI;(★ Enable Interrupts ★)
END Insert;
```

Łatwo zauważyć, że jest to mechanizm uniwersalny, bo tokeny mogą być umieszczone w buforze cyklicznym zarówno przez zwykłe zadania, jak i przez procedury obsługi przerwań. Bufor tokenów jest odpowiednikiem kolejki zadań gotowych w systemie wielozadaniowym.

Dowolne zadanie w systemie, w implementacji typ *Task*, jest procedurą z parametrem typu *Token*. Wszystkie występujące w systemie zadania zgrupowano w tablicy *TaskTab*.

```
TYPE Task = PROCEDURE(Token);
VAR TaskTab: ARRAY[0..Maxtask] OF Task;
```

Scheduler odzyskuje procesor w chwili zakończenia działania przez zadanie aktywne. Następnie analizuje bufor z tokenami i usuwa z niego token wskazywany przez indeks *OutSlot* oraz uruchamia zadanie (procedurę) przezeń wskazywane jako odbiorcze. Usuwany token *Tkn* jest przekazywany procedurze jako parametr. Gdy w pierścieniu brak aktywnych tokenów, to jest wykonywana procedura pusta *Dummy*. Opisany fragment kodu schedulera przedstawiono poniżej.

```
VAR InRing: CARDINAL; (★ Liczba aktywnych tokenów ★)
next  : CARDINAL;
  Tkn : Token ;
LOOP
  DI;(★ Disable Interrupts ★)
  IF  InpSlot >= OutSlot THEN InRing := InSlot - OutSlot
  ELSE InRing := RingSize + InpSlot - OutSlot  END;
  EI; (★ Enable Interrupts ★)
  IF InRing # 0 THEN (★ Istnieją tokeny nieobsłużone )
    Tkn := Ring[OutSlot]; next:= Tkn.To;
    TaskTab[next](Tkn); (★ Wywołanie procedury ★)
DI;
 OutSlot := (OutSlot + 1) MOD RingSize;
EI;
 ELSE(★ Brak tokenów nieobsłużonycch ★)
  Dummy
 END;
END;
```

W przedstawionym modelu uaktywnione zadanie *Z* musi wykonać się do końca, po czym jest wznawiany scheduler. Jeżeli zadanie *Z* potrzebuje niedostępnego w danej chwili zasobu, to może odroczyć swoje wykonanie umieszczając się na końcu kolejki, przez wykonanie procedury *Insert*(*Tkn*), przy czym pola tokenu *Tkn.From* i *Tkn.To* są równe identyfikatorowi *Z*. W trakcie wykonywania zadania przerwania są typowo odblokowane. Przerwania muszą pozostać zablokowane tylko na krótki okres wykonywania operacji na pierścieniu tokenów (jest to sekcja krytyczna).

Zaprezentowany mechanizm mimo prostoty oferuje w jednolity sposób dość szeroki zestaw usług wymaganych w systemach wielozadaniowych. Pojęcie tokenu i procedura *Insert* umożliwiają realizację następujących usług:

- synchronizację zadań,
- przekazywanie danych między zadaniami,
- uaktywnianie zadań przez procedury obsługi przerwań,
- opóźnianie zadań.

Mechanizm przekazywania tokenów jest podobny do wykorzystywanego w wielu systemach operacyjnych mechanizmu

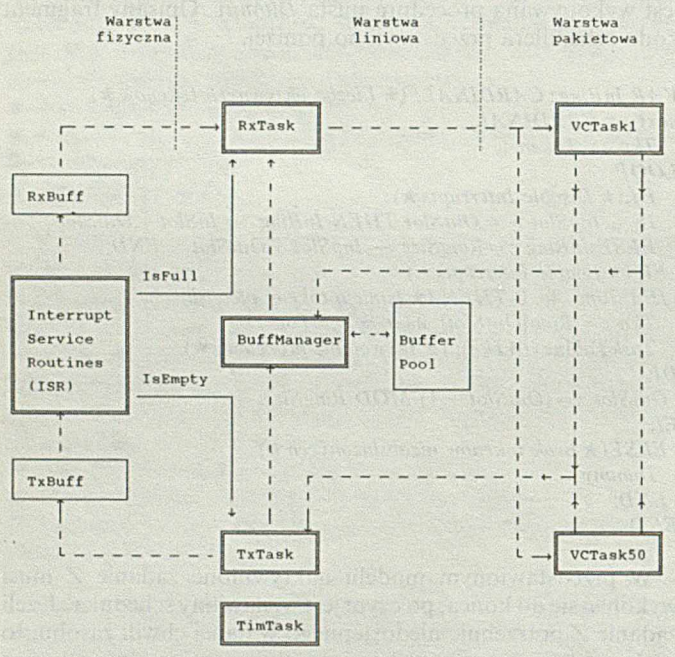
przekazywania wiadomości między zadaniami (ang. *message passing*) [1].

Istotną zaletą jest to, że czas przełączenia procesu jest zredukowany do rzędu czasu wywołania procedury, a czas zablokowania przerwań jest wielokrotnie mniejszy niż w systemie z schedulerem klasycznym.

Tę potencjalnie lepszą wydajność systemu z schedulerem transakcyjnym można zmierzyć budując modelowe oprogramowanie komunikacyjne. W opisywanym eksperymencie badaniu poddano szkieletowe oprogramowanie protokołu X.25 [6]. Oprogramowanie to zaimplementowano wykorzystując scheduler transakcyjny oraz, w celach porównawczych, system programowania współbieżnego, w tym przypadku TopSpeed Modula-2 [6].

Struktura oprogramowania protokołu X.25

Protokół X.25 jest protokołem 3-warstwowym, zatem w sposób naturalny jego implementacja może być zdekomponowana na zadania realizujące funkcje tych trzech warstw. Zadania te działają z natury asynchronicznie i środowisko, w którym są implementowane, musi zapewniać mechanizmy do ich synchronizacji i komunikacji. Szkieletową strukturę oprogramowania X.25 pokazano na rys. 2.



Rys. 2. Warstwy i przepływ danych w systemie

W warstwie fizycznej dla każdego portu komunikacyjnego rezydują dwie procedury obsługi przerwań (ISR): nadawania i odbierania. Operują one danymi nadawanymi (odbieranymi) (do) z właściwego dla kierunku transmisji bufora cyklicznego, tj. *TxBuff*(*RxBuff*). Procedury tej warstwy wymagają synchronizacji z warstwą liniową w dwóch przypadkach: po pierwsze wówczas, gdy z nadawczego bufora *TxBuf* zakończono nadawanie „w linię" pojedynczej ramki protokołu liniowego (a tym samym jest w nim miejsce na ewentualną następną ramkę do nadania), po drugie wówczas, gdy procedura odbiorcza skompletowała „z linii" pewną liczbę znaków, które tworzą razem jednostkę protokołu liniowego. Wymienione zdarzenia synchronizacyjne zaznaczono symbolicznie na rys. 2 jako *IsFull* i *IsEmpty*.

W warstwie liniowej rezydują cztery zadania:
RxTask – zadanie odbiornika,
TxTask – zadanie nadajnika,
BuffManager – zadanie zarządzania pulą buforów,
TimTask – zadanie timera.

Zadanie *RxTask* odbiera ramki od warstwy fizycznej kopiując je do buforów otrzymanych z puli wolnych buforów (przez zadanie *BuffManager*). Ramki typu *U* i *S* są przez zadanie *RxTask* przetwarzane lokalnie i (lub) we współpracy z zadaniem *TxTask*. Pozostałe, tj. *I*-ramki, są przekazywane do odpowiedniego zadania rezydującego w warstwie pakietowej.

Zadanie *TxTask* wpisuje odpowiedni nagłówek do wszystkich buforów z pakietami odbieranymi od zadań warstwy pakietowej. Po zbudowaniu *I*-ramek są one kopiowane do bufora *TxBuf* w warstwie fizycznej. Po przekopiowaniu do bufora nadajnika *TxBuf*, bufory są znaczone jako wolne i zwracane zadaniu *BuffManager*. Dodatkowo, przetwarzając informacje od zadania *RxTask*, zadanie *TxTask* wstrzymuje (wznawia) przepływ *I*-ramek do warstwy fizycznej. Zadanie *BuffManager* nadzoruje przydział buforów. Wykrywa stany niskiego (wysokiego) poziomu wolnych buforów w systemie. Przy osiągnięciu poziomu niskiego (wysokiego) *BuffManager* rozsyła stosowną informację do wszystkich zadań będących potencjalnymi konsumentami wolnych buforów. Zadanie *TimTask* odlicza czas na żądanie dowolnego zadania w systemie.

W warstwie trzeciej rezydują zadania utrzymujące połączenia wirtualne. Testowano wersję oprogramowania, w której było 50 takich połączeń. Każde takie zadanie, oznaczone *VCTask*, przetwarza bufor z *I*-ramką. Po przetworzeniu bufor taki zostaje zwolniony (oddany zadaniu *BuffManager*) bądź alternatywnie zadanie *VCTask* gereruje nowe dane i wówczas są one umieszczane w posiadanym przez zadanie buforze. Następnie bufor ten zostaje przekazany do dalszego przetwarzania do zadania *TxTask*. W ogólnym przypadku uaktywnienie dowolnego zadania *VCTask* następuje wskutek działań zadań *RxTask*, *TimTask* lub *BuffManager*.

Wykorzystanie schedulera transakcyjnego do implementacji protokołu X.25

Strukturę oprogramowania protokołu X.25 opisano powyżej. Jego implementacja przy użyciu schedulera transakcyjnego jest dość prosta. Uszczegółowienia wymaga jedynie sposób synchronizacji i przekazywania danych między zadaniami.

Procedura obsługi przerwań odbiornika umieszcza odbierane znaki w buforze *RxBuf* i wykrywa separatory jednostek protokołu. Po wykryciu jednostki procedura uaktywnia zadanie *RxTask* mające odebrać jednostkę z bufora. Uaktywnienie następuje przez umieszczenie w pierścieniu tokenu *Tkn1* = = (*IsFrame*, *ISR*, *RxTask*, *NIL*) za pomocą procedury *Insert*(*Tkn1*).

Pole *IsFrame* tokenu specyfikuje typ zdarzenia (pojawienie się jednostki). Pola *ISR* i *RxTask* specyfikują zadanie uaktywniające i uaktywniane. Pole *DataPtr* jest równe *NIL*, gdyż dane nie są przekazywane. Gdy token *Tkn1* zostanie wybrany przez scheduler, to następuje uaktywnienie zadania *RxTask* z parametrem *Tkn1*. Zadanie *RxTask* rozpoznaje typ zdarzenia (*Tkn.Command* = *IsFrame*) i przystępuje do jego obsługi. Kopiuje jednostkę z *RxBuf* do pobranego wcześniej bufora *TmpBuf* i przetwarza ją. Gdy ramka jest przeznaczona dla zadania *VCTask* poziomu pakietowego, zadanie umieszcza w pierścieniu token *Tkn2* = (*GetFrame*, *RxTask*, *VCTask*, *TmBuf*). W tym przypadku po wybraniu tokenu *Tkn2* scheduler uruchomi zadanie *VCTask*, przekazując mu jednocześnie dane – *TmpBuf*.

Bardziej szczegółowy opis wszystkich zadań wykracza poza ramy tego artykułu. We wszystkich jednak przypadkach zasada synchronizacji i komunikacji przez przekazywanie tokenów jest zachowana.

Implementacja oprogramowania komunikacyjnego w systemie wielozadaniowym

Aby umożliwić analizę porównawczą opisany model zaimplementowano przy użyciu języka programowania współbieżnego – Top Speed Moduli-2 [7]. Wykorzystano dostarczany z pakietem priorytetowy scheduler wywłaszczający. Synchronizacja zadań odbywała się przy wykorzystaniu semaforów i specjalnie zaimplementowanych skrzynek komunikacyjnych (ang. *mailbox*) [1].

Eksperymenty pomiarowe

W przeprowadzonych eksperymentach mierzono trzy parametry:

- przepustowość systemu,
- jałowy czas procesora,
- czas przełączenia kontekstu.

Eksperymenty wykonano na mikrokomputerze PC AT z zegarem 10 MHz [2]. Wyniki dla obydwu implementacji zebrano w tabelach 1 i 2.

Tabela 1. Implementacja z schedulerem transakcyjnym

Szybkość (Kb/s)		2,4	4,8	9,6	19,,2	38,4	57,6
Przepustowość		1	1	1	1	1	0,7
Jałowy czas CPU (%)	minimalny	86	79	76	31	0	0
	średni	90	83	79	36	7	0

Tabela 2. Implementacja w systemie wielozadaniowym

Szybkość (Kb/s)		2,4	4,8	9,6	19,2	38,4	57,6
Przepustowość		1	1	1	–	–	–
Jałowy czas CPU (%)	minimalny	25	20	16	–	–	–
	średni	28	27	29	–	–	–
Czas przełączania kontekstu = 1580 μs (1,58 ms)							

Jak widać z powyższych zestawień, zastosowanie schedulera transakcyjnego pozwala osiągnąć szybkość transmisji 57,6 Kb/s, a więc ponad pięć razy większą niż przy użyciu uniwersalnego systemu wielozadaniowego (9,6 Kb/s). Wynik taki nie powinien zaskakiwać. W kodzie schedulera transakcyjnego tylko na bardzo krótkie odcinki czasowe są zablokowane przerwania (operacje na pierścieniu tokenów). W modelu wielozadaniowym natomiast przerwania są zablokowane na czas przełączenia kontekstu oraz operacji na kolejkach i innych zmiennych systemowych (25 razy dłużej). Także narzut na operacje systemowe, np. szeregowanie, jest znacznie większy. Przy szybkości 57,6 Kb/s i zastosowanym procesorze czasy obsługi przerwań są na granicy wydolności systemu, w wyniku czego przy „zgubieniu" przypadkowych znaków następują retransmisje całych ramek i tym samym rejestrowana przepustowość względna spada do poziomu 0,7.

* * *

Wykonane eksperymenty jednoznacznie wskazują na zalety schedulera transakcyjnego. Umożliwia on uzyskanie pięciokrotnie większej szybkości transmisji niż z zastosowaniem uniwersalnego systemu wielozadaniowego. Nie bez znaczenia jest także prostota implementacji, łatwość śledzenia, mały rozmiar kodu (ok. 3 KB) i łatwa przenośność schedulera transakcyjnego. W istocie wystarczającym narzędziem do implementacji systemu jest kompilator języka wysokiego poziomu. Czyni to proponowane podejście szczególnie atrakcyjnym w implementacji autonomicznych sterowników komunikacyjnych, mogących pracować bez systemu operacyjnego. Rolę systemu operacyjnego pełni tu scheduler. Ceną efektywności schedulera transakcyjnego jest brak możliwości wywłaszczenia zadania bieżącego. Dlatego niedopuszczalne jest wykonywanie długotrwałych, potencjalnie przerywalnych zadań „w tle", np. operacji dyskowych. Reasumując, prezentowany scheduler transakcyjny wydaje się być szczególnie atrakcyjny dla specjalizowanych sterowników komunikacyjnych i wbudowanych systemów czasu rzeczywistego.

LITERATURA

[1] Bemmerl T., Schoeder G.: A portable Real-Time Multitasking Kernel for Embedded Microprocessors Systems. Microprocessing and Microprogramming, No. 21, pp. 181–188, 1987

[2] Fruźlewicz Z., Ułasiewicz J.: Transaction Scheduling for Communications Software Purposes. Proc. of International Conference – Computer Networks, Wrocław 1991

[3] Grothe D.: X.25 Software: Transaction Scheduling Keys Efficiency. Digital Design, December 1985, pp. 44–49

[4] iRMX-96 Operating System Reference Manual. Intel Corporation. 3065 Bowers Avenue. Santa Clara ((CA) 95051

[5] QNX Manual: Quantum Software Systems Ltd. Canada, 1990

[6] Recommendation X.25. CCITT Blue Book, Vol. VIII, Geneva 1989

[7] Top Speed Modula-2. User's Manual. Jensen & Partners International, 1990.

Współpraca Intel – IBM

W połowie listopada ub.r. przedstawiciele firm Intel oraz IBM zawarli porozumienie dotyczące współpracy w dziedzinie rozwoju procesorów dla przyszłych generacji komputerów osobistych. Porozumienie zawarto na okres 10 lat, a jego realizacja rozpocznie się już w tym roku przez wkład Intela w postaci znacznie ulepszonego pod względem funkcjonalnym procesora i486.

Pierwszych konkretnych wyników współpracy koncernów w dziedzinie opracowania nowych konstrukcji oczekuje się dopiero w 1993 r. Będzie to już całkowicie nowy procesor, łączący w ramach jednej kostki o architekturze układu 486 technologię grafiki XGA firmy IBM, pamięć typu cache oraz sterowniki pamięci i szyny systemu. Nad projektem tym będzie pracował wspólny zespół złożony z ok. 100 pracowników naukowych, zatrudnionych w wyodrębnionym ośrodku badawczo-rozwojowym. Budowę i wyposażenie tego ośrodka sfinansują w równych częściach obaj partnerzy porozumienia.

Dzięki porozumieniu Intel spodziewa się osiągnąć istotne zmniejszenie gwałtownie rosnących nakładów na prace rozwojowe, natomiast IBM – tak potrzebną mu przewagę nad coraz groźniejszą konkurencją przy wprowadzaniu nowych generacji komputerów. Poprzedni rozwój konstrukcji IBM opierał się wyłącznie na modyfikowaniu układów Intela, co zapewniało zawarte porozumienie licencyjne. Przykładem tej kończącej się już drogi rozwoju jest przygotowany ostatnio przez IBM układ 386-SLC, będący zmodyfikowaną wersją intelowskiego układu 80386SX.

JUNISOFTEX Sp. z o.o.

44-100 GLIWICE ul. Konstytucji 11,

tel.-faks 31-75-10, 31-90-81 do 88 w. 250, 272, 282, teleks 036233

JUNISOFTEX – to firma z tradycjami i najdłuższymi doświadczeniami w eksploatacji wielodostępnych systemów komputerowych Novell w kraju.

JUNISOFTEX – to najstarszy w kraju wykonawca własnych wielodostępnych systemów pracujących w sieci NetWare firmy Novell.

JUNISOFTEX – to autoryzowany reseller amerykańskiej firmy Novell.

JUNISOFTEX – to dostawca zintegrowanych systemów komputerowych działających w kilkudziesięciu firmach obejmujących w każdym przedsiębiorstwie, niezależnie od formy własności i dziedziny gospodarki: finanse, majątek obrotowy, majątek trwały i nietrwały, sprzedaż, techniczne przygotowanie produkcji oraz kadry-płace.

JUNISOFTEX – to dostawca i wykonawca autoryzowanych sieci lokalnych firmy Novell.

JUNISOFTEX – to dostawca sprawdzonego sprzętu komputerowego renomowanych firm ALR, IBM, TEAM, DIGILAB, EPSON, MANNESMANN TALLY.

JUNISOFTEX – to wyłączny i autoryzowany dystrybutor komputerów i terminali firmy DIGILAB na Śląsku.

JUNISOFTEX – to dostawca i wykonawca okablowania lokalnych sieci komputerowych ARCNET, ETHERNET na kablach zwykłych i światłowodowych oraz na dwużyłowych kablach telefonicznych w przypadku stosowania oprogramowania NetWare ACCESS SERVER firmy Novell.

JUNISOFTEX – to nauczyciel, który chętnie podzieli się swoją wiedzą na organizowanych kursach w swojej szkole informatycznej – w tym nauczy Cię: podstaw informatyki, obsługi sprzętu komputerowego, eksploatacji własnych systemów informatycznych, edytorów tekstu: CHiWRITER, WordPerfect, arkuszy kalkulacyjnych QPRO v. 3.0, LOTUS 1-2-3 itp.

JUNISOFTEX – to doradca w zakresie komputerowych metod organizacji i eksploatacji systemów komputerowych.

JUNISOFTEX – to strażnik postępu i nowoczesności w Twojej firmie.

JUNISOFTEX – to gwarancja niezawodności, rzetelności i terminowości.

JUNISOFTEX – to Twój doradca i partner, któremu możesz zaufać, który Cię nigdy nie zawiedzie.

JUNISOFTEX – to partner, który Cię wysłucha i zawsze pomoże podjąć dobrą decyzję.

Jeżeli chcesz

- ☆ zastosować najnowocześniejsze systemy komputerowe oraz lokalne sieci komputerowe,
- ☆ zreorganizować, usprawnić i szybko skomputeryzować swoje przedsiębiorstwo,
- ☆ lepiej wykorzystać już istniejące w Twoim przedsiębiorstwie komputery,
- ☆ wymienić swoje niefortunnie zakupione oprogramowanie,
- ☆ pozbyć się problemów eksploatacyjnych, upadających systemów, nie działających komputerów,
- ☆ przeszkolić załogę swojego przedsiębiorstwa,
- ☆ dobrze zainwestować swoje pieniądze i podnieść rangę swojego przedsiębiorstwa,
- ☆ w przyszłości mieć dostęp do krajowej i światowej sieci komputerowej,

TO ZGŁOŚ SIĘ DO NAS!

0/17/91

DARIUSZ ZONENBERG
Instytut Informatyki Teoretycznej i Stosowanej
Zespół Mikroprogramowania
PAN
Gliwice

Emulator układowy mikroprocesorów jako narzędzie uruchomieniowe w ujęciu programistycznym

Bardzo często korzystanie z narzędzia uruchomieniowego, jakim jest emulator mikroprocesora, bywa ograniczone przez nieznajomość istoty jego pracy i możliwości jakie on oferuje. Od strony użytkowej, emulator stanowi narzędzie uruchomieniowe prototypowych systemów mikroprocesorowych, mające cechy analizatora stanów logicznych oraz klasycznego programu uruchomieniowego (debuggera), rozszerzone o dodatkowe możliwości aktywnego wpływu na przebieg procesu uruchamiania. W przypadku korzystania ze stereotypowych metod uruchamiania systemu mikroprocesorowego, użytkownik jest zmuszony do korzystania z wielu narzędzi nawzajem się uzupełniających. Są to kolejno: analizator stanów logicznych, skrośne oprogramowanie narzędziowe danego mikroprocesora, symulator mikroprocesora oraz jego zasobów i opcjonalny debugger rezydujący w pamięciach stałych (ROM) uruchamianego systemu, wymagający często dodatkowej elektroniki dla wizualizacji wyników uruchamiania oprogramowania. Czasami wykorzystuje się również emulator lub symulator pamięci PROM, pozwalające na diagnostykę i aktualizacje docelowego programu sterującego. Oferowane instalacje emulatorów układowych mikroprocesorów mają z reguły cechy wszystkich wymienionych powyżej narzędzi, co znacznie upraszcza zarówno instalację uruchomieniową, jak i jej użytkowanie.

Zwykle opis funkcjonalny pracy emulatora jest równoważny opisowi syntaktyki i semantyki jego zleceń, zamieszczonemu w dokumentacji użytkowej i technicznej. Dokumentacje te są jednak z reguły podporządkowane idei łatwości korzystania z nich, często uporządkowane alfabetycznie lub tematycznie i nie mogą stanowić ogólnego informatora lub podręcznika użytkowania opisywanego sprzętu. Wyczerpująca informacja podana w dokumentacji użytkowej często zbyt komplikuje ogólną zasadę pracy i wykorzystania narzędzia, co w porzypadku niedoświadczonego użytkownika tylko odstręcza od wniknięcia w szczegóły wykorzystania licznych możliwości emulatora układowego.

W niniejszym artykule dokonano próby zdefiniowania funkcji emulatora w publikacyjnej wersji języka Pascal. Zakłada się, że większość programistów, nawet pracujących w języku asemblera, spotkało się z Pascalem, a w wielu przypadkach był on ich podstawowym narzędziem programowym w czasie nauki oprogramowania maszyn cyfrowych. Emulator jest narzędziem sprzętowym śledzącym pracę mikroprocesora i realizującym złożone algorytmy współpracy z procesorem uruchamianego prototypu. Wydaje się celowe użycie do opisu jego funkcji języka algorytmicznego, którego semantyka oddaje także przebieg procesu, zamiast prostej gramatyki lub równań, oddających raczej konstrukcję niż funkcje urządzenia. Arbitralne przyjęcie określonego języka jest usprawiedliwione jego popularnością. Chociaż istnieją języki lepiej nadające się do tego celu, ich mniejsze rozpowszechnienie wymagałoby dodatkowych opisów składni, co zwiększyłoby rozmiary opracowania, nie wnosząc wielu nowych elemetów. Zresztą, samo tworzenie języków efektywnie opisujących emulację, sterowanie nią lub jej definiowanie, jest odrębną dziedziną, przekraczającą ramy poruszanego tutaj tematu.

Założenia dla języka opisu emulatora układowego

Aby uniknąć nieporozumień związanych z organizacją pamięci i danych emulowanych mikroprocesorów oraz z wewnętrzną organizacją języka Pascal, przyjęto kilka poniższych założeń:

- Maksymalna wartość liczby całkowitej języka jest większa niż maksymalna wartość adresu i danej mikroprocesora. Zakłada się, że takie typy danych, jak bity i bajty, są zdefiniowane w implementacji języka i mają odpowiednie nazwy: *Bit* i *Byte*.
- Zakłada się typową dla współczesnych mikroprocesorów bajtową organizację pamięci operacyjnej oraz dostęp do tej pamięci na poziomie bajtu, aby zmniejszyć liczbę definiowanych funkcji. Jak wiadomo, z bajtów można składać słowa 16-, 32-bitowe i dłuższe. Wystarczy tylko znać ich organizację dla danego mikroprocesora.
- Zakłada się organizację portów procesora jak w poprzednim punkcie.
- Mikroprocesor znajdujący się pod kontrolą emulatora może znajdować się albo w trybie pracy w prototypie (GO), albo w trybie pełnienia usług dla potrzeb emulatora, będąc przełączanym na przemian z magistrali uruchamianego prototypu na

Mgr inż. DARIUSZ ZONENBERG ukończył w 1985 r. Wydział Informatyki na Politechnice Śląskiej w Gliwicach o specjalizacji Oprogramowanie i Metody Informatyki. Od 1985 r. pracuje w Instytucie Informatyki Teoretycznej i Stosowanej PAN w Gliwicach (do 1989 r. ZSAK PAN). Brał udział w budowie oprogramowania podstawowego mikrokomputera ComPAN-16, rodziny emulatorów RTDS/2-PC (emulacja układowa procesorów 8086, 80186, 80286, 80386, 8051). W ostatnich latach zajmował się oprogramowaniem sterowników opartych na mikrokontrolerze 8051 (WI-20, WACS-31). Zainteresowania – systemy czasu rzeczywistego, formalna definicja oprogramowania.

magistralę wewnętrzną emulatora (MONITOR). Sposób zmiany trybu zależy od klasy mikroprocesora, a często informacja o jego realizacji jest ściśle chronioną tajemnicą producenta. Ogólna definicja nagłówków procedur przełączania trybu może być następująca:

```
procedure Get_CPU; (★ Procedura przejścia w tryb MONITOR ★)
procedure Put_CPU (Flag: integer);
                    (★ Procedura przejścia w tryb GO, parametr Flag
ma następujące wartości:
                        RUN – praca ciągła,
                        BRK – praca ciągła z detekcją programowych
                              punktów kontrolnych,
                        TRP – wykonanie jednego rozkazu mikroprocesora.  ★)
```

Parametr BRK oznacza, że jest wykorzystywany naturalny mechanizm programowych punktów kontrolnych mikroprocesora, często uzyskiwany przez zamianę kodu programu na ekstrakod (zwykle jednobajtowy) napotkania punktu. Mechanizm ten, wykorzystywany przez większość debuggerów, nie jest konieczny do pracy emulatora, ale często się go wykorzystuje jako dodatkową możliwość zatrzymania realizacji programu w interesującym operatora punkcie. Programowe punkty kontrolne z reguły nie mogą być ustawione w pamięci stałej (ROM), ale za to ich liczba może być praktycznie nieograniczona i praca z nimi jest dla programisty wygodna i naturalna. Parametr TRP zwykle odnosi się do naturalnego w mikroprocesorze lub nadzorowanego przez emulator pobrania i wykonania jednego rozkazu procesora.

Kanał komunikacyjny emulatora układowego

Jak już wspomniano, emulowany procesor może pracować w dwu podstawowych trybach. Tryb nazwany GO jest naturalnym trybem pracy mikroprocesora w uruchamianym układzie prototypowym i mikroprocesor pracuje wtedy w czasie rzeczywistym, ze swoją naturalną szybkością, mogąc być równocześnie śledzonym przez emulator. Przejście z trybu GO w tryb MONITOR polega na sprzętowym upozorowaniu sytuacji, w której procesor w naturalny dla sieci sposób, przez wykonanie skoku, przerwania lub ekstrakodu wymuszonego przez emulator, rozpoczyna realizację programu Monitora, rezydującego w pamięci emulatora i adresowalnego w pamięci operacyjnej emulowanego mikroprocesora w dowolnie lub arbitralnie wybranym obszarze adresowym. Po zachowaniu rejestrów i słowa stanu mikroprocesor realizując program Monitora sprawdza, czy od strony systemu emulatora nie zostały ustawione parametry żądania obsługi; jeśli tak, to wykonuje żądaną operację. Komunikacja z emulatorem następuje z reguły przez tzw. kanał komunikacyjny emulatora dostępny równocześnie zarówno od strony emulatora, jak i od strony emulowanego mikroprocesora (mogą to być pewne porty lub pewien obszar pamięci). Opisany w języku Pascal, program monitora ma następującą strukturę:

```
    procedure Monitor;
label
    Start;
var Parametr1, Parametr2,.., ParametrN : integer; (★ parametry ★)
    Command                            : integer; (★ zlecenie ★)
    procedure Execute (ID:integer);
    begin
        (★ Treść procedury obsługi zleceń od emulatora ★)
    end;
    begin
Start:                          (★ Miejsce wejścia po przerwaniu GO ★)
    Command := 0;               (★ „0" oznacza nic nie rób          ★)
    While Command=0;            (★ Oczekiwanie na zlecenie emulatora ★)
    Execute (Command);          (★ Obsługa zlecenia emulatora       ★)
    goto Start                  (★ Powrót do pętli oczekiwania      ★)
    end
```

przy czym komórki *Parametr1, Parametr2, .., ParametrN, Command* są ustawiane przez emulator wykorzystujący mechanizm kanału komunikacyjnego, procedura *Excute* jest ogólną procedurą dostępu do zasobów emulowanego systemu prototypowego, a powrót do trybu *GO* następuje poprzez odpowiednią wartość parametru *Command* i polega na odzyskaniu zapamiętanej zawartości rejestrów i słowa stanu oraz powrocie do przerwanego programu prototypu według zapamiętanej w trakcie przejścia do trybu *MONITOR*, lub narzuconej w trybie *MONITOR*, wartości licznika rozkazów.

Od strony emulatora dostęp do zasobów emulowanego systemu polega na wywołaniu zdefiniowanej uprzednio procedury *Get_CPU* (przełączanie w tryb *MONITOR*), wykonywanej z reguły sprzętowo, oraz uaktywnieniu właściwego zlecenia (*Command*) wraz z odpowiednimi parametrami. Uproszczona procedura ustawiania parametrów i kodu operacji od strony emulatora może mieć następującą postać:

```
begin
    SetParms;                   (★ Ogólne ustawienie parametrów      ★)
    Command := ⟨liczba⟩         (★ Ustawienie kodu operacji          ★)
    while not (Command=0)
            ;                   (★ Oczekiwanie na wykonanie          ★)
    GetAnswer                   (★ Ogólne pobranie wyników przez kanał
                                   komunikacyjny                     ★)
end
```

Zakładając, że podstawowe zasoby systemu mikroprocesorowego to zawartość rejestrów, pamięci i ewentualnych portów, otrzymuje się trzy pary podstawowych operacji, które można wykonać po wywołaniu procedury *Get_CPU*, tzn.:

```
(★ Pobranie i ustawienie zawartości rejestrów ★)
    function Get_reg(Register:REGS):integer;
    procedure Put_reg(Register:REGS, Value:integer);
(★ Pobranie i ustawienie zawartości pamięci operacyjnej ★)
    function Get_mem(Address:integer):Byte;
    procedure Put_mem(Address:integer, Value:Byte);
(★ Pobranie i ustawienie zawartości portów ★)
    function Get_port(Address:integer):Byte;
    procedure Put_port(Address:integer, Value:Byte);
```

oraz jedną dodatkową, już opisaną procedurę powrotu do kontekstu GO – Put_CPU(Flag:integer). Użyty powyżej typ danych *REGS* jest typem wyliczeniowym zawierającym nazwy wszystkich rejestrów emulowanego mikroprocesora.

Realizacja zleceń debuggera oprogramowania

Operując zdefiniowanymi powyżej procedurami dostępu do zasobów mikroprocesora można otrzymać zestaw zleceń, jakimi dysponuje większość debuggerów:

A. Wyświetlanie i modyfikacja zawartości rejestrów (*Get_reg, Put_reg*).

B. Dostęp do pamięci:

- wyświetlanie zawartości pamięci w dowolnym formacie danych (*Get_mem*),
- modyfikacja zawartości pamięci (*Get_mem, Put_mem*),
- wypełnianie pamięci wzorcem (*Put_mem*),
- załadowanie pamięci danymi lub programem (*Put_mem*),
- przesuwanie zawartości pamięci (*Get_mem, Put_mem*),
- wyszukiwanie w pamięci wzorca (*Get_mem,*),
- asemblacja/disasemblacja (*Put_mem, Get_mem*).

C. Zapis i odczyt zawartości portów (*Put_port, Get_port*).

D. Uruchamianie programów:

- w pracy ciągłej (*Put_CPU*(*RUN*)),
- po rozkazie (*Put_CPU*(TRP)),

● w pracy ciągłej z punktami kontrolnymi kodu programu (użycie *Put_mem* w celu ustawienia ekstrakodu punktu oraz *Put_CPU*(*BRK*); oczywiście należy uprzednio zapamiętać zawartość zapisywanej ekstrakodem komórki, pobierając ją przez *Get_mem*).

W przypadku użycia powyższych możliwości emulatora, emulacji pamięci stałych na uruchamianej prototypowej karcie oraz zrealizowania możliwości współpracy z nazwami symbolicznymi użytymi w postaci źródłowej języka programującego emulowany system prototypowy, otrzymujemy narzędzie równie wygodne, jak symulator procesora na dużym systemie komputerowym oraz mające takie zalety małego monitora uruchomieniowego rezydującego w pamięci stałej (ewentualnie emulowanej) rzeczywistego systemu mikroprocesorowego, jak realność otoczenia mikroprocesora oraz rzeczywiste warunki pracy programu. Dodatkową zaletą emulatora (dotyczy to tylko emulatorów dwumagistralowych, w których faza MONITOR jest sterowana i kontrolowana przez odrębny procesor) w stosunku do debuggera wykorzystującego ten sam procesor jest fakt, że podobnie jak w dużym systemie komputerowym z symulatorem, konsola jest niezależna od biegu programu i w przypadku zapętlenia lub załamania oprogramowania jest możliwa ingerencja użytkownika, w celu sprawdzenia w jakim stanie znajduje się mikroprocesor. Zarówno symulator, jak i emulator mają z reguły własny system programowy (nazywany także często systemem bazowym), nadrzędny w stosunu do symulowanego programowo lub emulowanego sprzętowo mikroprocesora, pozwalający na aktywną reakcję ze strony operatora w przypadku każdego podejrzenia o błąd w uruchamianym programie.

Praca w trybie analizatora magistrali

Niezależnie od opisanych możliwości dostępu przez operatora do zasobów mikroprocesora, podstawowym problemem w czasie uruchamiania prototypowego systemu mikroprocesorowego jest poprawność pracy magistrali oraz zgodność oprogramowania czasu rzeczywistego z założonymi algorytmami i uzależnieniami czasowymi. Sprzętowe śledzenie przez emulator sygnałów układu scalonego procesora pozwala identyfikować jego stany w poszczególnych cyklach magistrali. W ogólnym przypadku program emulatora ma wgląd w wartość wystawianego adresu, danej oraz statusu magistrali o następującej strukturze:

```
type Bus = record
             Address: integer;
             Data   : integer; (★Dana 8-, 16- lub 32-bitowa ★)
             Status: array [0..N-1] of Bit;
           end;
var CPU_Bus: Bus;
```

przy czym N jest liczbą bitów słowa stanu magistrali. Uruchamiając współpracę z magistralą, użytkownik może używać emulatora jako analizatora stanów logicznych magistrali, analizując opisaną powyżej strukturę. Operator ma wgląd w rzeczywisty stan magistrali oraz zna wartości rzeczywistych danych i kodu, jakie z pamięci operacyjnej zostały przez magistralę odczytane, a znając wystawiane adresy jest w stanie odczytane dane porównać z wartościami, jakie powinny być odczytane. Emulator – jak każdy analizator stanów – jest wygodnym narzędziem diagnozującym zwarcia między poszczególnymi sygnałami magistrali. Analizę stanów można rozszerzyć także do sygnałów nie należących do struktury *Bus* (przez analizę dodatkowych sygnałów prototypu wybranych przez użytkownika spoza podstawki mikroprocesora w uruchamianym systemie). W większości przypadków daje to dostateczne przesłanki do wnioskowania o przyczynach potencjalnych błędów konstrukcji. W większości konstrukcji emulatorów istnieje również możliwość pracy po cyklu maszynowym (np. przez wstrzymanie procesora w stanie WAIT, sztucznie forsowanym), co pozwala na bieżąco analizować magistralę, a w cyklach odczytu z magistrali nawet podstawiać własne wartości danych.

Emulatory często mają pamięci, w których można przechować określoną liczbę cykli magistrali, co w zestawieniu z możliwością analizy słowa stanu daje dodatkową możliwość przeglądania stanów magistrali pod kątem realizowanego przez mikroprocesor oprogramowania, wykonywanego w czasie rzeczywistym i w uruchamianym środowisku. Przykładowo, jest możliwa disasemblacja danych pobieranych jako kod, drukowanie histogramów efektywności dostępu do danych programu itp. Zapamiętany w ten sposób ślad stanów magistrali może być przetwarzany w najróżniejszy sposób, pod kątem wspomnianej już realizacji programu, zmian słowa stanu, współpraca ze sterownikami urządzeń zewnętrznych, jest także możliwa jego różnoraka wizualizacja – od binarnej, przez symboliczną aż do graficznej, zależnie od potrzeb i obiektów, którymi operator jest zainteresowany.

Istnieją mikroprocesory mające między modułem sterowania magistrali a modułem wykonywania rozkazów kolejkę, przechowującą pobrane, lecz jeszcze nie wykonane, bajty rozkazów, związane z trudnością jednoznacznego odniesienia sekwencji stanów magistrali do sekwencji stanów zasobów wewnętrznych mikroprocesora. Niektóre z nich (np. Intel 8086) zawierają w słowie statusu magistrali informację o kolejce rozkazów, inne jednak nie wyprowadzają informacji o kolejce na zewnątrz i próba śladowania ich magistrali oraz wyciągania ze śladu efektywnych wniosków napotyka liczne trudności lub nawet jest obarczona prawdopodobieństwem błędów interpretacji (np. Motorola 68000). Ogólnie biorąc analiza śladu polega na przetworzeniu następującej tablicy:

```
Trace_memory: array [0..M-1] of Bus;
```

gdzie M jest maksymalną możliwą liczbą stanów magistrali, jakie można zapisać.

Śledzenie stanów magistrali ma jeszcze jeden dodatkowy aspekt. Często zdarza się, że program, na skutek nie wykrytych jeszcze błędów programowych lub sprzętowych, powoduje skok w obszar adresowy, w którym nie ma pamięci, lub nagle zaczyna modyfikować swój kod lub stałe dane. Dzięki możliwości śledzenia wystawianego przez procesor adresu i jego słowo stanu, w wielu konstrukcjach emulatorów istnieje możliwość deklaracji zakazanych lub nakazanych obszarów pamięci dla dostępów z określonym słowem stanu. Funkcje te są z reguły realizowane przez specjalnie zaprojektowane moduły sprzętowe. Dodatkowo, w ramach sprzętowego przetwarzania adresu, często stosuje się tzw. emulowanie pamięci operacyjnej – dostarczanie pamięci operacyjnej emulowanemu mikroprocesorowi w zamian za jego pamięć (np. w celu wyeliminowania pamięci stałych prototypu) lub dostarczanie mu dodatkowej pamięci (np. zamiast opcjonalnej karty pamięci, z którą docelowo powinien współpracować). Emulsja pamięci zapewnia dodatkową kontrolę pracy emulowanego mikroprocesora, gdyż umożliwia formułowanie od strony emulatora właściwości pamięci lub nawet wykorzystywanie jej jako dwubramowej, w celu obserwowania wartości danych.

Mając możliwość śledzenia sygnałów wejściowych i wyjściowych mikroprocesora, można według określonych kryteriów dokonywać prób ich automatycznej oceny. W prostych przypadkach dotyczy to przykładowo poprawnego poziomu zasilania lub nieaktywności sygnału zerowania.

Dla sygnałów magistrali, tzn, adresu, danej i słowa stanu wprowadza się w konstrukcji emulatora specjalny moduł testo-

wania tych wartości na zgodność z wartościami zadanymi. Testy dotyczą wykrywania określonego statusu lub ich sumy logicznej oraz wartości adresu i danej. W typowej relacji adresu i danej testy obejmują większość, mniejszość, zawieranie się w przedziale, równość wartości na magistrali w stosunku do wartości zadanych lub zgodność z zadeklarowaną maską bitów. Istnieje również możliwość wykrywania całych sekwencji stanów magistrali. Dla uproszczenia notacji założono, że ustawienie warunku realizuje procedura:

procedure Set_condition(*Condition*: *Cond*);

gdzie typ danej *Cond* został zadeklarowany następująco:

```
type Cond = record
              (★ Szczególna zawartość deklaracji warunku będąca najczęściej odbiciem struktury sprzętowej modułu testującego ★)
            end;
```

a opis sposobu deklaracji warunku zawiera się w parametrze *Condition*, którego struktura nie zostanie opisana, aby zachować ogólność rozważań. W celu sprawdzenia, czy warunek jest spełniony, zostanie zdefiniowana funkcja logiczna

function Test_condition(*State*: *Bus*): *Boolean;*

sprawdzająca, czy stan magistrali *State* spełnia warunek ustawiony przez *Set_condition*. Funkcja ta jest wyzwalana i realizowana sprzętowo w każdym cyklu maszynowym po jej odblokowaniu.

Mając możliwość testowania spełnienia określonego warunku przez zawartość magistrali, większość emulatorów realizuje dwie podstawowe klasy reakcji, uzależnionych od spełnienia warunku – sprzętowego punktu zatrzymania oraz selektywnego śladowania.

Sprzętowy punkt zatrzymania programu

Pierwszą podstawową klasą reakcji na spełnienie warunku na magistrali jest przerwanie realizacji programu sterującego prototypu i wyjście emulowanego mikroprocesora do trybu *MONITOR*, aby umożliwić operatorowi wgląd w zasoby systemu mikroprocesorowego. Testowanie stanu magistrali w celu sprawdzenia warunku przerwania pracy jest zwykle nazywane sprzętowym punktem zatrzymania. W porównaniu z programowym punktem kontrolnym, który dotyczy tylko dostępu do adresu realizowanego programu, sprzętowy punkt zatrzymania dotyczy dostępu do adresu kodu lub danej, wartości odczytywanej (zapisywanej) danej oraz słowa stanu magistrali, w którym dany cykl wystąpił. Można go także ustawić w pamięci każdego rodzaju (tzn. i ROM, i RAM), podczas gdy programowy punkt kontrolny mógł być ustawiony tylko w pamięci RAM. Niezależnie od zalet sprzętowego punktu kontrolnego ma on swoje ograniczenie wynikające z konieczności rozbudowy wyszukanych układów wykrywających zgodność ze wzorcem. Uwzględniając możliwość deklaracji sprzętowego punktu kontrolnego, otrzymujemy następującą konfigurację parametrów procedury *PUT_CPU:*

RUN – praca ciągła;

TRP – wykonanie jednego rozkazu procesora z automatycznym wywołaniem procedury *Get_CPU* po zakończeniu;

SBR – praca ciągła z detekcją ekstrakodu punktu kontrolnego, po jego napotkaniu jest realizowana procedura *Get_CPU*,

HBR – praca ciągła ze sprzętową detekcją warunku sprzętowego na magistrali; po spełnieniu warunku jest realizowana procedura *Get_CPU*,

SHB – praca ciągła z detekcją ekstrakodu punktu kontrolnego i z równoczesną detekcją warunku sprzętowego na magistrali, po napotkaniu punktu programowego lub spełnieniu warunku sprzętowego jest realizowana procedura *Get_CPU*.

Dodatkowo, po użyciu parametrów *RUN, SBR, HBR, SHB* emulator ze swojej strony może wywołać funkcję *Get_CPU*, zależnie od wymagań operatora.

Kwalifikator śladowania magistrali

Drugą klasą reakcji na spełnienie warunku testowanego na magistrali jest uaktywnienie kwalifikatora śladowania. Chodzi tu o wybiórcze zapisywanie do pamięci śladu magistrali (*Trace-memory*) tylko określonych cykli. Typowe przypadki wybiórczego zapisu są następujące:

a) wykonanie zapisu śladu od momentu spełnienia warunku (np. w celu zaśladowania wykonania procedury),

b) wykonanie zapisu śladu tylko w momencie spełnienia warunku (np. w celu śledzenia dostępów do komórki pamięci lub portów),

c) wykonanie zapisu śladu wokół momentu spełnienia warunku, na *K* cykli przez i po spełnieniu warunku (np. w celu otrzymania disasemblacji kodu operującego na danej),

d) wykonanie zapisu śladu od momentu spełnienia jednego warunku, w cyklach wokół momentu, w którym wystąpił drugi albo od spełnienia jednego warunku do spełnienia drugiego (praca w ramkach śladu),

e) inne, szczególne sytuacje, w których odpowiedni dobór warunku śladowania może dać nieklasyfikowaną powyżej sytuację, a będzie jedynym właściwym sposobem wykrycia błędu programisty lub elektronika

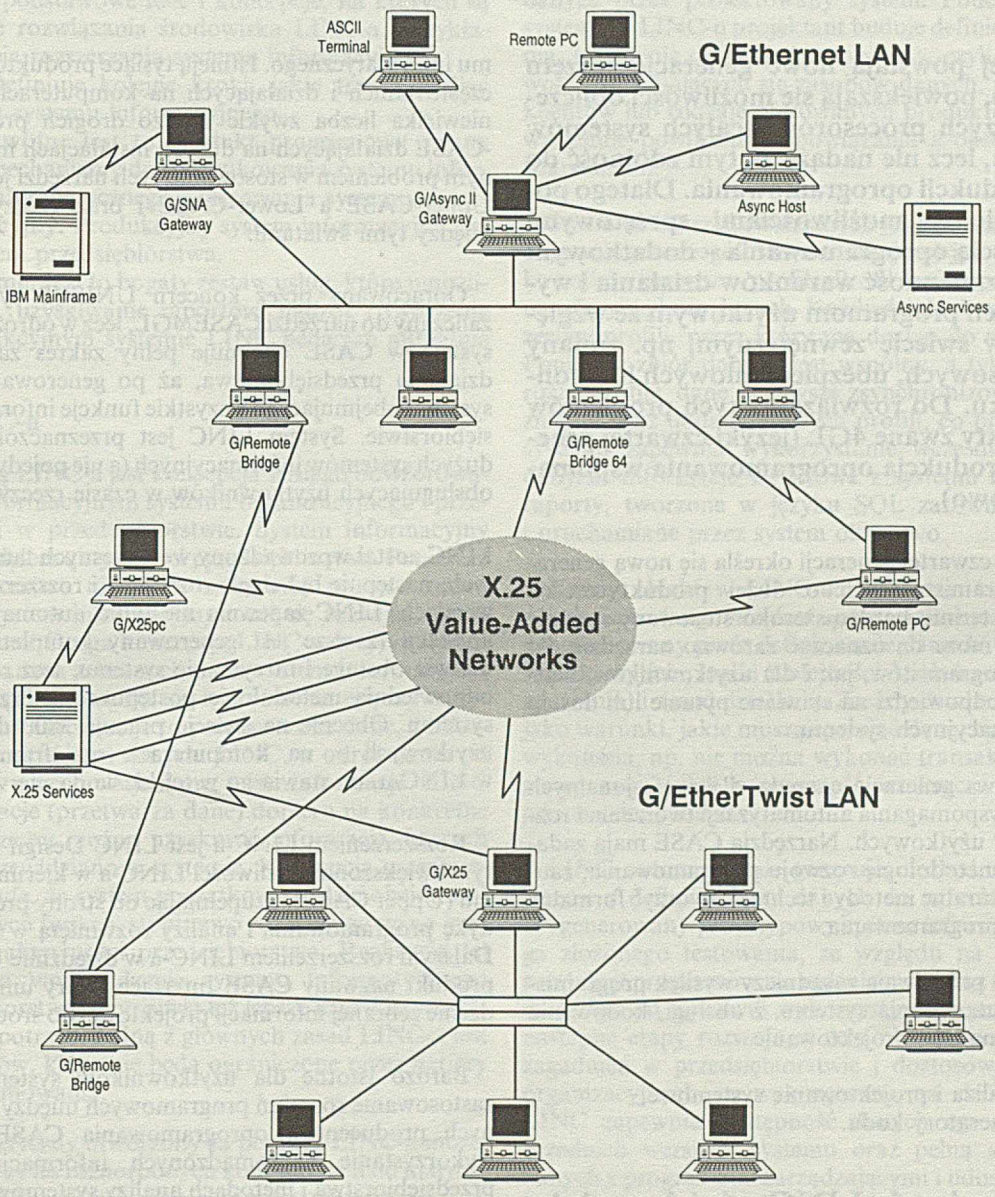
Emulatory większości napotykanych generacji, oprócz warunków na wartości występujące na magistrali, wykrywają także wartości kilku sygnałów, które sam operator może dostarczyć układom emulatora. Mogą to być dowolne sygnały z dowolnego źródła, spełniające tylko określone wymagania co do napięcia i mocy, pełniące funkcję zewnętrznego kwalifikatora śladowania lub sprzętowego punktu zatrzymania. Umożliwia to współpracę emulatora z miernikami elektrycznymi, oscyloskopami, analizatorami stanów lub innymi zewnętrznymi urządzeniami wspomagającymi uruchamianie. Zwiększa się w ten sposób integracja konfiguracji uruchomieniowej, gdyby zaszła konieczność jej rozbudowy o bardziej specjalizowane układy diagnostyki elektroniki uruchamianej prototypowej karty mikroprocesorowej. Podobną rolę odgrywają także czasami zintegrowane z oprogramowaniem emulatora pakiety wizualizacji wyników pomiarów wymienionych urządzeń na konsoli systemu bazowego emulatora.

Szczegółowe materiały na temat opisanego emulatora można uzyskać kontaktując się bezpośrednio z Autorem artykułu: **mgr inż. Dariusz Zonenberg, Instytut Informatyki Teoretycznej i Stosowanej PAN, Gliwice, tel. 31-08-11 wew. 194 i 195.**

Przypominamy!
Regularne otrzymywanie
INFORMATYKI
może zapewnić
tylko prenumerata!

Produkty sieciowe LAN i WAN

Gateway communications, inc.



BEZKONKURENCYJNE PRODUKTY
SIECI KOMPUTEROWYCH NAGRADZANE PRZEZ:

PC MAGAZINE, LAN MAGAZINE, INFO WORLD

OFERUJE AUTORYZOWANY DYSTRYBUTOR:

MIKROB

P.W.P.T. MIKROB SP. Z O.O.

20-346 Lublin, ul. Długa 5
Tel. (0-81) 420-61, faks 415-43, teleks 643776 MIKRO PL

Na życzenie wysyłamy katalog produktów. Atrakcyjny program współpracy dla dealer'ów.

0/15/91

Artykuł został przygotowany na III Górską Szkołę PTI, która odbyła się w Szczyrku w dniach 17–21 czerwca 1991 r.

ROMAN CUPISZ
UNILOT sp. z o.o.
Warszawa

LINC – kompletne środowisko CASE

Coraz szybciej powstają nowe generacje sprzętu komputerowego, powiększają się możliwości obliczeniowe pojedynczych procesorów i całych systemów komputerowych, lecz nie nadąża za tym zdolność do zwiększania produkcji oprogramowania. Dlatego powstaje luka między możliwościami sprzętowymi a produktywnością oprogramowania – dodatkowym problemem jest zmienność warunków działania i wymagań stawianych programom użytkowym ze względu na zmiany w świecie zewnętrznym, np. zmiany przepisów finansowych, ubezpieczeniowych lub konkurencji na rynku. Do rozwiązania tych problemów powstały produkty zwane 4GL (języki czwartej generacji) i CASE (produkcja oprogramowania wspomagana komputerowo).

Terminem języki czwartej generacji określa się nową generację narzędzi programistycznych o dużej produktywności. W praktyce jest to termin bardzo szeroko stosowany i różnie rozumiany, tak że może on oznaczać zarówno narzędzie dla profesjonalnych programistów, jak i dla użytkowników, ułatwiające uzyskanie odpowiedzi na stawiane pytanie lub dostęp do zasobów informacyjnych systemu.

CASE jest to nowa generacja narzędzi dla profesjonalnych programistów, do wspomagania automatyzacji tworzenia i rozwijania programów użytkowych. Narzędzia CASE mają zadanie dyscyplinować metodologię rozwoju programowania, zautomatyzować strukturalne metody i techniki, tworzyć formalne systemy rozwoju oprogramowania.

Narzędzia CASE przesuwają zasadniczy wysiłek programistów z tworzenia i utrzymania systemu, z obsługi, kodowania i testowania – na analizę i projektowanie.
Używane terminy:
Upper-CASE – analiza i projektowanie systemowe,
Lower-CASE – generatory kodu.

Jakie miejsce zajmuje LINC wśród produktów CASE?

Produkty określane mianem CASE wykonują liczne zadania, które można podzielić na następujące grupy:
- analiza potrzeb przedsiębiorstwa,
- analiza systemowa,
- projektowanie systemu,
- kodowanie programów,
- integracja i testowanie,
- utrzymanie systemu (obsługa i konserwacja).

Produkty CASE znajdujące się na rynku są zwykle przeznaczone do obsługi małego fragmentu cyklu rozwojowego systemu informatycznego. Istnieją tysiące produktów Upper-CASE, często tanich i działających na komputerach klasy PC, oraz niewielka liczba zwykle bardzo drogich produktów Lower-CASE działających na dużych instalacjach mainframe. Głównym problemem w stosowaniu tych narzędzi jest obszar między Upper-CASE a Lower-CASE i brak rzeczywistej integracji między tymi światami.

Opracowany przez koncern UNISYS system LINC jest zaliczany do narzędzi CASE/4GL, lecz w odróżnieniu od innych systemów CASE obejmuje pełny zakres zadań, od analizy działania przedsiębiorstwa, aż po generowanie kompletnego systemu obejmującego wszystkie funkcje informacyjne w przedsiębiorstwie. System LINC jest przeznaczony do tworzenia dużych systemów informacyjnych (a nie pojedynczych aplikacji) obsługujących użytkowników w czasie rzeczywistym.

LINC został wprowadzony we wczesnych latach osiemdziesiątych, następnie był ciągle rozwijany i rozszerzany w kolejnych wersjach. LINC zapewnia nie tylko automatyczną integrację aplikacji (zawsze jest generowany kompletny system) oraz ułatwia obsługę i utrzymanie systemu, lecz również zapewnia odpowiednią metodologię postępowania przy projektowaniu systemu. Obecnie na świecie pracuje 3800 dużych systemów użytkowych – na komputerach mainframe – napisanych w LINC-u, co stawia go w tej klasie na pierwszym miejscu.

Rozszerzeniem LINC-a jest LINC Design Assistent, w którym zwiększono możliwości LINC-a w kierunku analizy systemu (Upper CASE), uzupełniając od strony programowej metodykę programowania i analizy rozwiniętą w ramach LINC-a. Dalszym rozszerzeniem LINC-a w dziedzinie Upper CASE jest produkt nazwany CASE Interface, który umożliwia wprowadzenie zebranej informacji projektowej do środowiska LINC-a.

Bardzo istotne dla użytkowników systemów CASE jest zastosowanie sprzężeń programowych między produktami różnych producentów oprogramowania CASE, co umożliwia wykorzystanie nagromadzonych informacji o organizacji przedsiębiorstwa i metodach analizy systemowej. Duże znaczenie ma możliwość stosowania w systemie CASE operacji zwanych reverse engineering i re-engineering.

- **Reverse Engineering** umożliwia zbudowanie definicji z bieżącego systemu fizycznie zainstalowanego, co czyni konserwację i modyfikację systemu łatwiejszą i bardzie efektywną.
- **Re-engineering** jest to aktualizacja istniejącego systemu, słowników danych, bazy danych (restrukturalizacja). W wyniku otrzymuje się nowy system, który potrafi wykonać wszystko to, co robił system doychczasowy. Zmniejsza to wysiłek obsługi systemu (aktualizacji) i jednocześnie w przyszłości pozwala umieścić program użytkowy w nowym środowisku sprzętowo-programowym. LINC zawiera obie te funkcje, co znacznie zwiększa efektywność posługiwania się tym systemem.

Co to jest LINC?

Termin LINC jest trudny do zdefiniowania, gdyż w zależności od poziomu – warstwy znaczeniowej – określa się tym mianem różne pojęcia. Ogólnie można wyróżnić następujące znaczenia:

- filozofia,
- metodologia,
- środowisko rozwoju,
- system informacyjny,
- usługi Run-Time.

Filozofia. Są to podstawowe idee i koncepcje, na których są oparte konkretne rozwiązania środowiska LINC-a. Przykładem jest koncepcja rozszerzania systemu informacyjnego.

Metodologia. Definiuje kolejne kroki, jakie trzeba podjąć w celu rozwijania systemu informacyjnego.

Środowisko rozwoju. Jest to produkt programowy zwany LINC. Zawiera mechanizmy do specyfikowania systemu informacyjnego oraz automatycznego generowania systemu.

System informacyjny. Produkcyjny system informacyjny jest po prostu modelem przedsiębiorstwa.

Usługi RUN-Time. Jest to bogaty zestaw usług, które umożliwiają sterowanie, uzyskiwanie raportów, nadzór i tworzenie zapytań w produkcyjnym systemie LINC będącym aktualnie w użyciu.

Filozofia LINC-a

Zasadniczą ideą LINC-a jest koncepcja ścisłego odwzorowania w systemie informacyjnym systemu organizacyjnego i przepływu informacji w przedsiębiorstwie. System informacyjny powinien być modelem aktywności przedsiębiorstwa. Ponieważ żadne przedsiębiorstwo nie może zatrzymać procesu zmian i rozwoju, więc system informacyjny musi się rozwijać razem z przedsiębiorstwem. Druga koncepcja LINC-a dotyczy sposobu traktowania danych zebranych przez system informacyjny; najogólniej mówiąc wprowadza się rozróżnienie między danymi, a informacjami. Informacje to przetworzone dane, z informacji nie można odtworzyć danych. LINC, w odróżnieniu od innych systemów, przechowuje dane, a nie informacje. LINC dostarcza informacje (przetwarza dane) dopiero na konkretne żądanie, dzięki czemu można uzyskiwać informacje, których pierwotnie nie przewidziano w systemie. Koncepcja rozszerzenia systemu zakłada, że system początkowo będzie obejmował kilka podstawowych funkcji, stopniowo rozciągając się na inne działy i obszary aktywności przedsiębiorstwa. Realizacja tej koncepcji ułatwia wprowadzenie systemu informatycznego w życie przedsiębiorstwa, umożliwia też lepsze jego dostosowanie do bieżących potrzeb. Jedną z głównych zasad LINC-a jest tworzenie systemów, które nie będą ograniczone przez bariery sprzętu i oprogramowania.

System informatyczny utworzony za pomocą LINC-a może:

- być wykonywany na różnorodnych komputerach od małych, np. typu microA, do największych systemów mainframe;
- komunikować się w prostych i złożonych sieciach LAN oraz sieciach rozległych WAN;
- komunikować się z innymi systemami LINC-a, systemami UNISYS-a innymi niż LINC oraz systemami innych producentów.

Metodologia LINC-a

Zasadniczym punktem jest ukierunkowanie na potrzeby przedsiębiorstwa. Oznacza to naturalną zależność między elementami w projektowaniu LINC-a, a środowiskiem organizacyjnym, dla którego jest projektowany system użytkowy. LINC jest środowiskiem umożliwiającym projektowanie objektowe (jest to najnowsza technologia w programowaniu i projektowaniu systemów).

Przedstawiamy teraz oczywiście w sposób bardzo skrócony, sposób projektowania systemów w LINC-u. W pierwszym kroku określa się strukturę przedsiębiorstwa i jego potrzeby w zakresie systemu informatycznego oraz wybiera kilka działów powiązanych w naturalny posób przez obieg informacji. Następnie należy zidentyfikować podstawowe operacje (transakcje), które są przeprowadzane w tych działach; w terminologii LINC-a są one nazywane zdarzeniami. Przykładem mogą być takie transakcje, jak sprzedaż, uregulowanie rachunku, wysyłka towaru. Każde z tych zdarzeń wymaga aktualizacji pewnych danych przez projektowany system. Podczas projektowania systemu w LINC-u projektant buduje definicję każdego zdarzenia. Następnie definiuje komponenty, czyli zestawy względnie wolno zmieniających się zasobów danych, np.: lista adresowa klientów lub rodzaje oferowanych produktów, oraz ustala ich wzajemne powiązania i powiązania z określonymi wcześniej zdarzeniami.

Strumień zdarzeń (widzianych jako paczki danych odpowiadających poszczególnym transakcjom) jest dostępny dla użytkowników przez tzw. profile. Profil można porównać z tradycyjną relacyjną bazą danych. Każdy dział lub użytkownik ma swój zestaw profili, przez który ma dostęp do strumienia zdarzeń. Ponieważ zbiór danych jest wspólny, a tylko oglądany przez różne profile, dane są zawsze zsynchronizowane. Jest możliwa zmiana oraz dodanie nowych profili, co przy modyfikowaniu systemu zapewnia wykorzystanie wcześniej zgromadzonych danych. Informacje wyjściowe z systemu otrzymujemy przez raporty, tworzone w języku SQL zarówno na bieżąco, jak i uruchamiane przez system okresowo.

Każdy obiekt w LINC-u: zdarzenie, komponent, ogólnie zwany ispec (ang. *interface specification*) jest następnie mikroprogramowany przez definiowanie danych, ich charakterystyki wejścia-wyjścia, logiki (kolejności działań związanych z obsługą zdarzenia) i zasad działania. Zasady działania są tu rozumiane jako warunki, jakie muszą spełniać dane, by transakcja została wykonana, np. nie można wykonać transakcji wysyłki towaru z magazynu, jeżeli jego wartość przekracza wysokość kredytu, jaki ma dany klient.

LINC automatycznie generuje niezbędne oprogramowanie łącznie z definicją bazy danych do przechowywania danych. Wygenerowany prototypowy system informatyczny nie wymaga złożonego testowania, ze względu na zautomatyzowanie produkcji programów zarządzających i baz danych. System ten może być od tej chwili wykorzystywany przez użytkowników; następne etapy rozwoju w celu objęcia systemem wszystkich zagadnień w przedsiębiorstwie i dostosowania się do zmian organizacyjnych są wykonywane na działającym systemie. LINC zapewnia dostępność danych zgromadzonych w poprzednich wersjach systemu oraz pełną synchronizację baz danych z programami zarządzającymi i udostępniającymi zasoby tych baz danych. Takie podejście zasadniczo skraca czas uzyskania efektu, tj. zysku z funduszy zainwestowanych w system informatyczny, oraz zmniejsza koszty obsługi i rozwoju systemu.

Implementacja środowiska LINC-a

Główne właściwości:

- filozofia projektowania ukierunkowana na potrzeby przedsiębiorstwa,
- metodologia rozwoju implementacji,
- interakcyjne środowisko rozwoju,
- współistnienie z innymi metodami i narzędziami (ma swoją metodę i narzędzie Design Assistent, ale współpracuje z innymi narzędziami Upper-CASE istniejącymi na rynku),

– kompletne generowanie systemu: automatyczne kodowanie, integracja i obsługa (utrzymanie) systemu (jest to znacząca różnica w porównaniu z innymi produktami).

Podczas projektowania są uwzględniane wszystkie elementy systemu. Każdy element jest jednocześnie dokumentowany i zawiera dokładną specyfikację, formaty ekranu, kontekstowe objaśnienia (ang. *help*) itd. Wszystkie aspekty dotyczące określonego obiektu są przechowywane w jednym miejscu, co ułatwia obsługę systemu i jego rozwijanie.

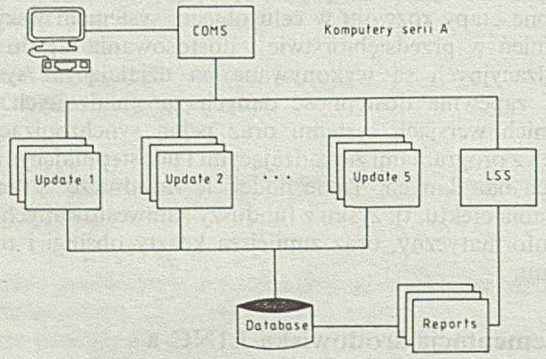
LINC Design Assistent jest narzędziem Upper-CASE wspomagającym analizę i projektowanie, działającym w środowisku graficznym. Podkreślenia wymaga fakt, że istnieje możliwość przedefiniowywania (ang. *reverse engineerinng*) ze specyfikacji LINC-a do modelu obiektowego Design Assistent z zachowaniem wszystkich informacji zawartych w specyfikacjach.

● **Interakcyjne funkcje projektowe:**
– interakcyjne projektowanie obrazu ekranu i raportu,
– definiowanie logicznej struktury,
– obsługa słownika danych: lokalnego (dla jednego systemu informatycznego) i globalnego (dla wielu systemów),
– definiowanie menu i dialogu z użytkownikiem.

● **Interakcyjne możliwości środowiska:**
– wybory z menu,
– pełna dokumentacja systemu LINC dostępna na bieżąco,
– interakcyjna kontrola składni,
– kontrola zgodności słownika danych: lokalnego i globalnego (automatyczna propagacja zmian).

Struktura systemu LINC

Strukturę systemu wygenerowanego za pomocą LINC-a przedstawiono na rys. 1; składa się on z wielu programów. Programy te są niewidoczne dla użytkownika, który może skoncentrować się na wprowadzaniu transakcji i formułowaniu zapytań. Kiedy użytkownik rozpoczyna sesję, musi podać swój kod i hasło. Dzięki temu ma on dostęp do systemu obsługi komunikatów (ang. *Message Control System* – MCS), znanego na komputerach serii A pod nazwą COMS. COMS formatuje i kieruje komunikaty przesyłane do (z) terminali. Obsługuje również współpracę z innymi systemami komputerowymi i przesyłanie komunikatów przez sieć.



Rys. 1. Przykładowa struktura systemu wygenerowanego za pomocą LINC-a

Programy nazywane Update obsługują zasadnicze funkcje systemu LINC. Database jest systemem baz danych zawierających strumień informacji przesyłanych przy rejestrowaniu poszczególnych zdarzeń. Logiczne funkcje systemu informacyjnego mogą być umieszczone w jednym programie Update lub podzielone funkcjonalnie między podsystemy. Podsystemy uruchamiają programy Update zawierające różne obiekty (zdarzenia, komponenty). Można uruchomić wiele wcieleń każdego programu Update. Z reguły najczęściej używane obiekty grupuje się razem w jednym podsystemie. Przykładowo, jeżeli w systemie bankowym jest wymagane wykonanie 50 tys. transakcji typu wpłata lub wypłata w godzinie szczytu i jednocześnie 50 transakcji typu obsługa depozytu lub hipoteki, to system uruchamia dziesięć wcieleń programu Update1, zawierającego obsługę wpłat i wypłat i tylko jedno wcielenie programu Update2 obsługującego pozostałe zlecenia.

Program LINC System Supervisor (LSS) zarządza wszystkimi programami, zapewniając ich synchronizację i spójność systemu. Programy Reports, korzystając bezpośrednio z bazy danych, produkują raporty okresowe i odpowiedzi na zapytania ad hoc.

Struktura systemu LINC (środowiska) jest analogiczna do systemu produkcyjnego.

Wymiana informacji z innymi systemami informatycznymi

Współpraca systemu informatycznego utworzonego w LINC-u z innymi systemami informatycznymi może, w zależności od potrzeb, dokonywać się na wiele sposobów:

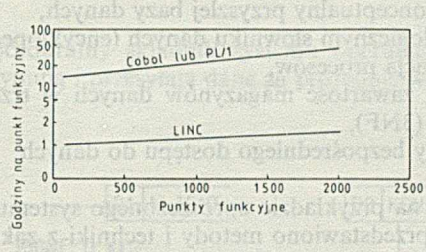
LSS – program zarządzający systemu LINC może obsłużyć żądania innego systemu LINC i sam takie zainicjować;
GLI (*Generalized LINC Interface*) – obsługuje strumień transakcji, formatując je automatycznie do postaci używanej przez LINC; każda transakcja musi zawierać opis formatu rekordu: może to służyć do aktualizowania danych i zapytań;
LINC USER – mechanizm umożliwiający inicjację transakcji z innym systemem użytkowym; LINC inicjuje program, który może być modyfikowany przez dodanie fragmentów napisanych przez użytkownika; wymagana jest odpowiedź do systemu LINC;
NOFORM – transakcje z systemem LINC z terminala lub programu; odpowiedź nie zawiera kodów kontrolnych i sterujących;
CALL – przekazuje parametry do podprogramu, który może być napisany w dowolnym języku dostępnym w środowisku danego komputera;
OFF-LINE – rekordy mogą być przesłane do LINC-a przez dowolny program, każdy rekord może być interpretowany przez interfejs OFF-LINE jako transakcja i przesyłany do działającego systemu użytkowego, tak jak by było to wprowadzane przez wewnętrzne wejście (terminal).

Porównanie wydajności LINC-a i Cobol-u lub PL/1

Określeniem produktywności systemu LINC interesuje się wielu ludzi, konkretne badania w tym kierunku podjął dr Rudolph, z University of Auckland w Nowej Zelandii. Opracowywał on w ramach IBM Share/Guide Group nowe techniki pomiaru funkcjonalności systemów użytkowych. Opublikowane przez niego rezultaty badan zostały obszernie zreferowane w książce J. Briana Dregera z Boeninga opublikowanej w 1989 pt. „Function Point Analysis". Technika ta jest obecnie dobrze znana i powszechnie akceptowana. Rezultaty badań są dzisiaj równie adekwatne jak wówczas, gdy zostały po raz pierwszy opublikowane.

Produktywność została zdefiniowana jako wysiłek konieczny do uzyskania określonych rezultatów, podzielony przez nakład czasu. Do określenia wysiłku przy tworzeniu i rozwijaniu oprogramowania dr Rudolph zastosował technikę punktów funkcyjnych.

Dla wielu projektów był obliczany czas niezbędny do zaprogramowania jednego punktu funkcyjnego. Wyniki pomiarów dla projektów realizowanych w LINC-u porównywano z rezultatami osiągniętymi dla Cobol-u i PL/1. Typowy czas dla LINC-a to 1 do 1,5 godziny na jeden punkt funkcyjny, a dla Cobol-u i PL/1 – odpowiednio 15 godzin, z tym że liczba ta znacznie rosła w miarę zwiększania się rozmiarów projektu. Projekt o wielkości 2000 punktów funkcyjnych wymagał więc nakładu już 50 godzin na jeden punkt funkcyjny. Dla porównania, w środowisku LINC-a zwiększenie wielkości projektu nie zmienia produktywności w tym samym stopniu (rys. 2).



Rys. 2. Porównanie wyników badań nad efektywnością produkcji oprogramowania z użyciem LINC-a oraz PL/1 lub Cobol-u (na osi pionowej – skala logarytmiczna)

Podsumowanie właściwości LINC-a

Nawet bardzo pobieżne omówienie wszystkich możliwości i aspektów stosowania LINC-a wymaga bardzo dużej objętości tekstu, dlatego ograniczę się do wymienienia najistotniejszych cech systemu LINC:

● **LINC generuje kompleksowy system informacyjny** (co ozacza objęcie przez programy wygenerowane przez LINC wszystkich funkcji obsługi bazy danych, komunikacji, obsługi użytkowników, zarządzania systemem użytkowym).

● **Dodatkowe możliwości:**
– automatyczne generowanie systemów rozproszonych,
– możliwość zdalnego i lokalnego wykonywania programów użytkowych,
– wymiana informacji między aplikacjami LINC-a,
– wykorzystanie inteligentnych stacji roboczych BTOS lub MS-DOS,
– współpraca (wymiana informacji) z innymi systemami, zarówno w trybie bezpośrednim, jak i wsadowym.

● **Szczególne właściwości:**
– natychmiastowa kontrola błędów syntaktycznych i niezgodności logicznych,
– kontrola słownika danych: lokalnego i globalnego,
– używanie najnowszej technologii CASE,
– bezpośrednie projektowanie raportów i wyglądu ekranu,
– automatyczna propagacja zmian,
– środowisko umożliwiające wprowadzenie zmian i rozwoju przez wielu programistów jednocześnie (w celu zwiększenia wydajności),
– możliwość przekazania sterowania do podprogramów napisanych w innych językach.

● **Korzyści ze stosowania LINC-a:**
– systemy nie są zależne od architektury komputera,
– systemy użytkowe można przenosić,
– jest przezroczysty dla wprowadzania nowych technologii,
– programy i dane są zawsze zsynchronizowane,
– jest możliwa automatyczna integracja składników systemu,
– czas potrzebny na testowanie systemu jest znacznie zredukowany.

● **Zaawansowane możliwości:**
– pełna kontrola nad wszystkimi raportami zarówno na ekranie terminala, jak i na drukarkach,
– specjalne możliwości dokumentacyjne systemu i nadzoru nad pracą systemu projektowego umożliwiające przesyłanie specyfikacji z jednego systemu do drugiego,
– zapytania ad hoc oparte na SQL-u z rozszerzeniami ułatwiającymi definicję zapytania,
– zapytania wbudowane w system,
– wielojęzyczność, do czterech języków jednocześnie,
– wielopoziomowy system bezpieczeństwa zarówno dla samego LINC-a, jak i dla systemu produkcyjnego,
– specjalne możliwości definiowania i generowania systemów pracujących w sieci, znacznie upraszczające definiowanie i zapewniające poprawność działania.

Specyfikacje LINC-a ze względu na wysoki poziom abstrakcji są niezależne od sprzętu i oprogramowania systemowego, co daje pełne możliwości przenoszenia systemów użytkowych. Obecnie LINC jest oferowany dla komputerów mainframe: A Series, V Series, 1100/2200 Series, System 80 oraz dla komputerów działających w systemie Unix.

Informacje z firmy Microsoft

PROTECH KUPUJE LICENCJĘ

16 grudnia ub.r. firma MICROSOFT podpisała z firmą PROTECH, uważaną za największego polskiego producenta komputerów osobistych, porozumienie licencyjne na wykorzystywanie najnowszej (wprowadzonej w czerwcu ub.r.) wersji 5. systemu operacyjnego MS-DOS. Od tego dnia wszystkie montowane i sprzedawane przez PROTECH komputery będą wyposażane wyłącznie w tę wersję systemu.

Obie firmy określają porozumienie jako decydujący krok na drodze zwalczania w Polsce piractwa w dziedzinie oprogramowania. W ocenie licencjodawcy MS-DOS jest obecnie najpowszechniej stosowanym na świecie systemem operacyjnym komputerów osobistych. Firma szacuje, że łączna liczba wykorzystywanych kopii tego systemu przekroczyła już 70 milionów. Microsoft z ubolewaniem jednak stwierdza, że niestety w wielu rejonach świata, w tym również w Europie Środkowej i Wschodniej, udział nielegalnych kopii tego systemu jest nadal znaczny.

Opóźnienie wersji 3.1 Windows

Nowa wersja 3.1 systemu operacyjnego Windows była zapowiedziana przez firmę Microsoft na koniec roku ubiegłego. Konieczność dalszego dopracowania ulepszeń oraz pracochłonność wszechstronnego przetestowania systemu spowodowały, że firma zadecydowała przesunąć termin jego przekazania do dystrybucji o jeden kwartał. W wyniku tej decyzji można oczekiwać, że kopie nowej wersji Windows pojawią się na rynku polskim dopiero późną wiosną br.

KATARZYNA STĄPOR
Akademia Ekonomiczna
Katowice

Strukturalna analiza i projektowanie systemów informatycznych – metodyka Gane-a – Sarsona (1)

Ogólną definicję metodyki tworzenia systemów informatycznych (TSI) podano w pracy [22], traktując ją jako spójny, logicznie uporządkowany zbiór metod i procedur o charakterze technicznym i organizatorskim pozwalających realizować cykl życia systemu informatycznego [10, 12] przez zespół wykonawczy. W pracy [22] sprecyzowano również poszczególne składniki metodyki TSI:

- modele opisu dziedziny przedmiotowej,
- strukturyzacja procesu TSI (procedury poszczególnych etapów),
- szczegółowe metody i techniki dokumentowania TSI,
- narzędzia wspomaganego komputerem TSI,
- specyfikacja wymagań merytorycznych wobec poszczególnych twórców,
- kryteria oceny, mechanizmy kontroli projektu,
- zasady planowania i sterowania rozwojem systemu.

Powyższe składniki poszczególnych metodyk stanowią wspólne środowisko analityków, projektantów i użytkowników w procesie TSI.

Zaproponowane przez Gane-a i Sarsona podejście do procesu TSI bywa nazywane **modelowaniem logicznym** [2, 6]. Opiera się w głównej mierze na opracowanej wcześniej przez Gane-a i Sarsona (oraz niezależnie przez Yourdona i deMarco [11]), strukturalnej analizie systemów [7], projektowaniu strukturalnym [23] oraz modelu związków encji ER (ang. *entity-relationship*) [1, 2], który stanowi podstawę licznych metodyk analizy i projektowania systemów. W pracach [5, 6] modelowanie logiczne przedstawiono jako proces przebiegający w siedmiu fazach:

1) opracowanie diagramów przepływu danych DFD (ang. *data flow diagrams*);
2) ustalenie logicznej zawartości magazynów danych;
3) zbudowanie na podstawie modelu ER modelu konceptualnego przyszłej bazy danych – diagramy ERD (ang. *entity-relationship diagrams*);
4) normalizacja magazynów danych na diagramach przypływu danych do trzeciej postaci normalnej (3NF);
5) aktualizacja diagramów przepływu danych (DFD) z wykorzystaniem struktur z kroku 3 i 4;
6) wydzielenie tzw. jednostek proceduralnych projektowanego systemu (pogrupowanie procesów i związanych z nimi danych w podsystemy fizyczne);
7) specyfikacja jednostek proceduralnych (modularny, hierarchiczny projekt oprogramowania).

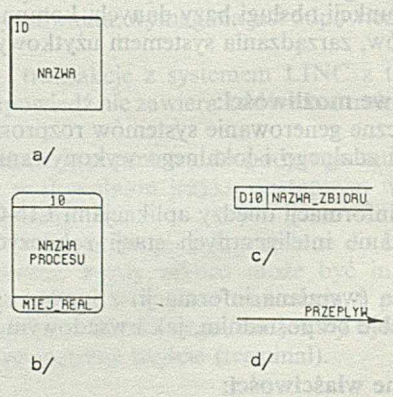
Etapy 6 i 7 służą transformacji uzyskanego we wcześniejszych etapach modelu logicznego na model fizyczny. Głównym celem modelowania logicznego jest uzyskanie logicznego modelu przyszłego systemu składającego się z następujących części:

- ogólne i szczegółowe diagramy przepływu danych,
- model konceptualny przyszłej bazy danych,
- dane w logicznym słowniku danych (encyklopedii systemu),
- specyfikacja procesów,
- logiczna zawartość magazynów danych w trzeciej postaci normalnej (3NF),
- diagramy bezpośredniego dostępu do danych.

Poniżej, na przykładzie uproszczonego systemu dystrybucji towarów, przedstawiono metody i techniki z zakresu modelu logicznego.

Diagram przepływu danych

Diagram przepływu danych jest podstawowym narzędziem modelowania logicznego w strukturalnej analizie systemów. Jest również wykorzystywany w pierwszym kroku projektowania strukturalnego. DFD stanowi sieciową reprezentację procesów (procedur, funkcji) w systemie oraz danych łączących te procesy. Ukazuje ogólny przepływ danych przez system, bez uwzględnienia czynnika czasu.



Rys. 1. Podstawowe składniki diagramu przepływu danych (notacja Gane-a-Sarsona):
a) obiekt zewnętrzny,
b) proces,
c) magazyn danych,
d) przepływ danych

Tworzą go cztery podstawowe elementy, których symboliczne oznaczenia przedstawiono na rys. 1.:

- przepływ danych,
- magazyn danych,
- proces,
- obiekt zewnętrzny.

● **Przepływ danych**
Opisuje przepływ (logiczny) strumienia danych o określonej zawartości między dwoma obiektami (źródłem i przeznaczeniem) na diagramie. Dozwolone są przepływy (w obie strony): obiekt zewnętrzny-proces oraz magazyn danych-proces.

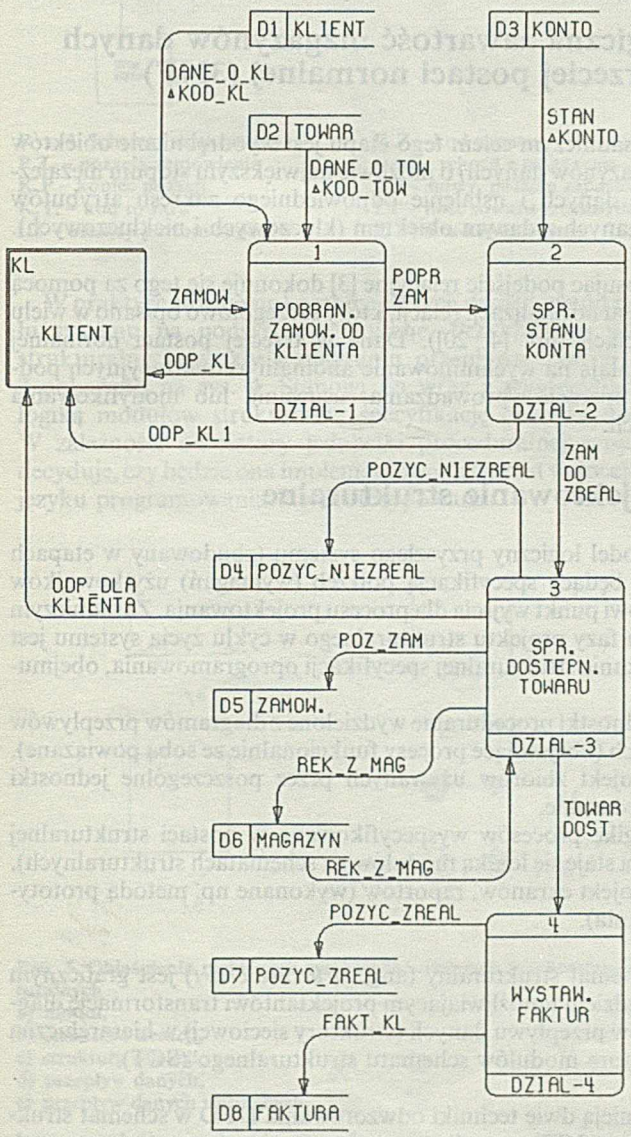
● **Magazyn danych**
Reprezentuje miejsce, w którym dane są przetrzymywane pomiędzy procesami, operującymi tymi danymi (jeśli występuje przesunięcie w czasie między tymi procesami). Przedstawia logiczny zbiór danych. Wyróżnia się globalne i lokalne magazyny danych. Globalne magazyny są umieszczone na ogólnym DFD lub jego rozwinięciach (wówczas na zewnątrz granicy rozwinięcia) i mogą być dzielone przez wszystkie diagramy rozwinięcia (magazyny *D*1 – *D*8 na rys. 2). Lokalne magazyny są umieszczone w obrębie granicy rozwnięcia na diagramie niższego poziomu (oraz na jego rozwinięciach).

● **Proces**
Stanowi proceduralny składnik systemu operujący danymi. Procesy otrzymują i przesyłają dane za pośrednictwem przepisów danych.

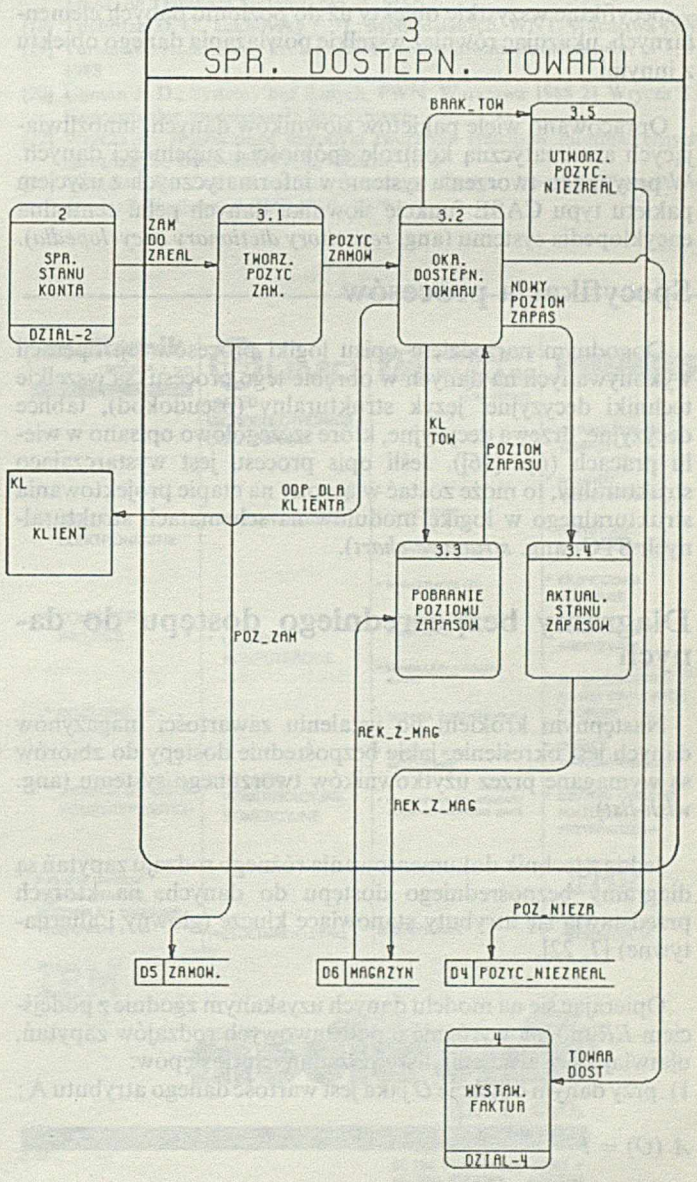
● **Obiekt zewnętrzny**
Reprezentuje źródło lub przeznaczenie informacji, która jest zewnętrzna w stosunku do systemu. Jest dostarczycielem informacji, będącej „bodźcem" dla pewnych procesów w systemie, lub odbiorcą informacji „produkowanej" przez system (np. KLIENT na rys. 2).

Rozróżnia się ogólne i szczegółowe diagramy przepływów danych. Ogólny DFD (rys. 2) jest diagramem najwyższego poziomu i przedstawia całościowy obraz tworzonego systemu, który następnie zostaje uszczegółowiony na diagramach niższego poziomu (szczegółowych DFD) – rys. 3. Diagram na rys. 3, będący rozwinięciem procesu 3 z ogólnego DFD na rys. 2 ukazuje szczegółowo subprocesy i przepływy składające się na realizację procesu 3. Postępując w podobny sposób można dalej rozwijać procesy poziomu drugiego w szczegółowe diagramy niższego poziomu. Ze względu na czytelność zaleca się, aby liczba procesów na jednym diagramie nie przekraczała 6–7 ([6]).

Ten proces definiowania systemu metodą zstępującą nazywa się rozwijaniem DFD (ang. *explosion*); daje w efekcie wielopoziomowe diagramy o wzrastającym stopniu szczegółowości.



Rys. 2. Ogólny diagram przepływu danych dla uproszczonego systemu dystrybucji towarów



Rys. 3. Szczegółowy diagram przepływu danych – rozwinięcie procesu 3 z rys. 2

Następnie dokonuje się weryfikacji utworzonego modelu procesów (tj. diagramów DFD), prowadząc analizę wejściowo-wyjściową procesów oraz magazynów danych na wszystkich diagramach DFD (bilans poziomy) oraz analizę wejściowo-wyjściową pocesów wraz z ich rozwinięciami (bilans pionowy). W przypadku posługiwania się narzędziami typu CASE czynności bilansowania są całkowicie zautomatyzowane.

Schemat konceptualny bazy danych

Na podstawie modelu danych związków encji (tzw. podejście obiekt-atrybut-związek [1, 2, 18]) dokonuje się opisu rzeczywistości (badanej dziedziny przedmiotowej) w postaci tzw. schematu (modelu) konceptualnego [18, 21], stanowiącego dokumentację jej własności logicznych. Zagadnienie modelowania danych oraz model danych związków encji zostanie omówiony w drugiej części artykułu.

Słownik danych (encyklopedia systemu)

Różnego rodzaju diagramy zawierają jedynie symboliczną reprezentację obiektów na nich występujących. Ich szczegółowa definicja powinna zostać umieszczona w centralnym słowniku danych. Definicja takiego obiektu ma strukturę hierarchiczną – specyfikuje wszystkie obiekty aż do poziomu danych elementarnych, ukazując również wszelkie powiązania danego obiektu z innymi.

Opracowano wiele pakietów słowników danych, umożliwiających automatyczną kontrolę spójności i zupełności danych. W przypadku tworzenia systemów informatycznych z użyciem pakietu typu CASE funkcję słownika danych pełni centralna encyklopedia systemu (ang. *repository dictionary encyclopedia*).

Specyfikacja procesów

Dogodnym narzędziem opisu logiki procesów (tj. operacji wykonywanych na danych w obrębie tego procesu) są wszelkie techniki decyzyjne: język strukturalny (pseudokod), tablice decyzyjne, drzewa decyzyjne, które szczegółowo opisano w wielu pracach (np. [16]). Jeśli opis procesu jest wystarczająco strukturalny, to może zostać włączony na etapie projektowania strukturalnego w logikę modułów na schematach strukturalnych STC (ang. *structure chart*).

Diagramy bezpośredniego dostępu do danych

Następnym krokiem po ustaleniu zawartości magazynów danych jest określenie, jakie bezpośrednie dostępy do zbiorów są wymagane przez użytkowników tworzonego systemu (ang. *wish-list*).

Jedną z technik dokumentowania różnego rodzaju zapytań są diagramy bezpośredniego dostępu do danych, na których przedstawia się atrybuty stanowiące klucze (główny i alternatywne) [7, 22].

Opierając się na modelu danych uzyskanym zgodnie z podejściem *ER* można wyróżnić 6 podstawowych rodzajów zapytań, ułatwiających ustalenie listy pożądanych dostępów:
1) przy danym obiekcie O jaka jest wartość danego atrybutu A :

$A\ (O) = ?$

2) przy danej wartości V danego atrybutu A podaj obiekty mające tę wartość:

$A\ (?)\ \{ = \# > < \}\ V$

3) dla danego obiektu O i danej wartości V podaj atrybuty odpowiadającego tej wartości:

$?\ (O)\ \{ = \# < > \}\ V$

4) dla danego obiektu O wyprowadź wartość wszystkich jego atrybutów:

$A\ (O) = ?$

5) podaj wartość danego atrybutu we wszystkich obiektach:

$A\ (?) = V$

6) przy zadanej wartości atrybutu wyprowadź wszystkie obiekty, w których atrybut ten ma zadane wartości:

$?\ (?)\ \{ = \# < > \}\ V$

Wszystkie zapytania mogą być wielokrotne, kwalifikowane. Ostatecznym efektem jest wyspecyfikowanie odpowiednich atrybutów kluczowych obiektów (kluczy głównych i alternatywnych).

Logiczna zawartość magazynów danych w trzeciej postaci normalnej (3NF)

Zasadniczym celem tego etapu jest wyodrębnienie obiektów (magazynów danych) o możliwie największym stopniu niezależności danych i ustalenie odpowiedniego zakresu atrybutów związanych z danym obiektem (kluczowych i niekluczowych).

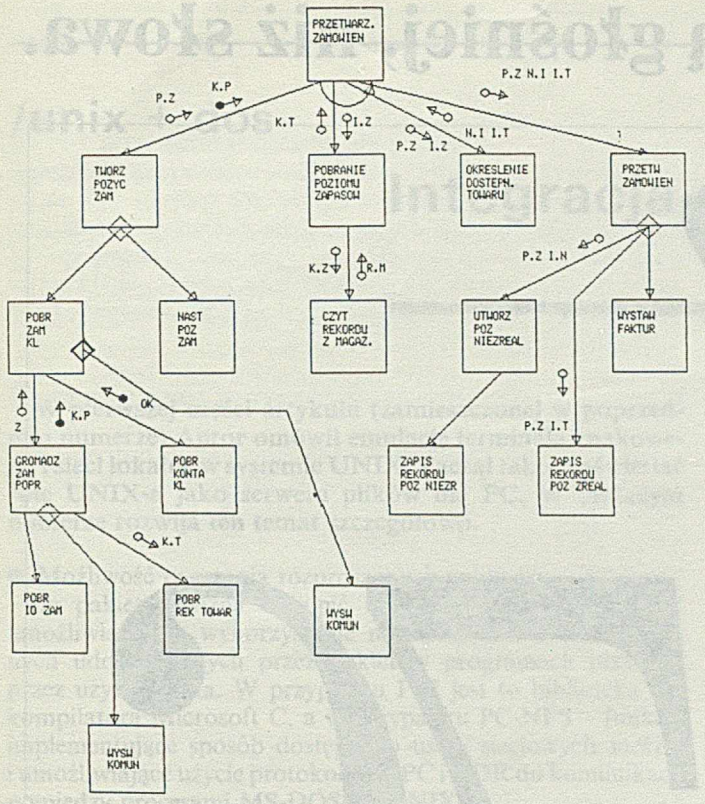
Stosując podejście relacyjne [3] dokonuje się tego za pomocą procesu normalizacji relacji, który szczegółowo opisano w wielu pozycjach (np. [4, 20]). Dane w trzeciej postaci normalnej pozwalają na wyeliminowanie anomalii aktualizacyjnych podczas operacji wprowadzania, usuwania lub modyfikowania danych.

Projektowanie strukturalne

Model logiczny przyszłego systemu (zbudowany w etapach 1–5), będący specyfikacją potrzeb (wymagań) użytkowników stanowi punkt wyjścia dla procesu projektowania. Zasadniczym celem fazy projektu strukturalnego w cyklu życia systemu jest uzyskanie strukturalnej specyfikacji oprogramowania, obejmującej:
- jednostki proceduralne wydzielone z diagramów przepływów danych (obejmujące procesy funkcjonalnie ze sobą powiązane),
- projekt zbiorów używanych przez poszczególne jednostki proceduralne,
- logikę procesów wyspecyfikowaną w postaci strukturalnej (która staje się logiką modułów na schematach strukturalnych),
- projekt ekranów, raportów (wykonane np. metodą prototypowania).

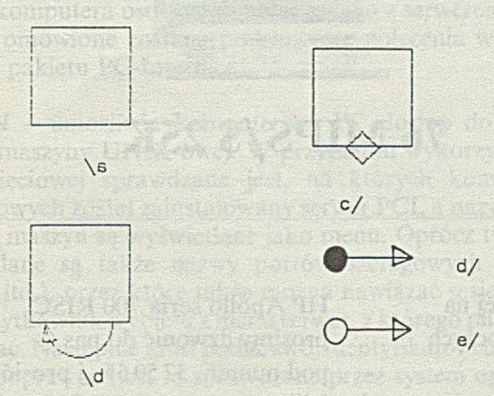
Schemat strukturalny (ang. *structure chart*) jest graficznym narzędziem umożliwiającym projektantowi transformację diagramów przepływu danych (struktury sieciowej) w hierarchiczną strukturę modułów schematu strukturalnego (SCT).

Istnieją dwie techniki odwzorowania DFD w schemat strukturalny (SCT) – analiza transformacyjna (przy podsystemach przetwarzania wsadowego) i transakcyjna (przy podsystemach przetwarzania bezpośredniego), które szczegółowo opisano np. w [14, 23].



Rys. 4. Schemat strukturalny
P.Z. – pozycja zamówienia
K.P – koniec pozycji
K.T. – kod towaru
I.Z – bieżący poziom zapasów
K.Z – kod zapasu
R.M – rekord z magazynu
N.I – nowy, poziom zapasów
I.T – ilość towaru zrealizowanego
Z – rekord zamówienia

W praktyce stosuje się kombinację tych dwóch metod podziału systemu na podsystemy fizyczne. Przykładowy schemat strukturalny powstały z diagramu przepływu danych z rys. 2 pokazano na rys. 4. Stanowi on wraz z wyspecyfikowaną logiką modułów strukturalną specyfikację oprogramowania. W zależności od natury jednostki proceduralnej projektant decyduje, czy będzie ona implementowana wprost w docelowym języku programowania lub prototypowana.



Rys. 5. Objaśnienia niektórych symboli stosowanych w schematach strukturalnych
a) moduł,
b) struktura iteracji,
c) struktura selekcji,
d) przepływ danych,
e) przepływ danych sterujących

LITERATURA

[1] Chen P. P.: Entity-Relationship Model – Towards a Unified View of Data. ACM Trans. on Database Systems, Vol. 1, No. 1, 1976
[2] Chen P. P.: The Lattice Structure of Entity Set. Proc. of the Fifth Int. Conf. on Entity--Relationship Approach, Dijon, France, 1986
[3] Codd E. F.: A Relational Model of Data for Large Shared Banks. Comm. of ACM, Vol. 13, No. 6, 1970
[4] Dutka A. F., Hanson H.: Fundamentals on Data Normalization. Addison-Wesley, 1989
[5] Fuglewicz P.: Systemy CASE – problemy, techniki, rozwiązania (1). Informatyka, nr 11–12, 1990
[6] Gane C.: Rapid System Development. Prentice-Hall, Englewood Cliffs (NJ), 1988
[7] Gane C., Sarson T.: Structured System Analysis – Tools and Techniques. Prentice-Hall, Englewood Cliffs (NJ), 1979
[8] Langefors B.: Infological Models and Information User Views. Inf. Syst., No. 5, 1980
[9] Lockemann P. C.: Data abstractions for database systems. ACM Trans. on Database Systems, No. 4, 1979
[10] Lucas H. C.: The Analysis, Design and Implementation of Information Systems. Prentice-Hall, (NJ), 1985
[11] de Marco T.: Structured Analysis and System Specification. Prentice-Hall, Englewood Cliffs (NJ) 1979
[12] Martin J.: Fourth Generation Languages. Prentice-Hall, Englewood Cliffs (NJ), 1985
[13] Martin F.: User-Centered Requirements Analysis. Prentice-Hall, Englewood Cliffs (NJ), 1988
[14] Martin J., McClure C.: Structured Techniques. Basis for CASE. Prentice-Hall Englewood Cliffs (NJ), 1988
[15] Page-Jones M.: The Practical Guide to Structured Systems Design. Yourdon Press/Prentice-Hall, Anglewood Cliffs (NJ), 1988
[16] Pollack S., Hicks M.: Tablice decyzyjne. PWN, Warszawa 1975
[17] Smith J. M.: Database abstractions: Aggregation and Generalization. ACM Trans. on Database Systems, No. 2, 1977
[18] Tsichritzis D. C., Lochovsky F. H.: Modele danych. WNT, Warszawa 1990
[19] Turski W.: Kanoniczny krok procesu programowaia. Informatyka, nr 8, 9, 1988
[20] Ullman J. D.; Systemy baz danych. PWN, Warszawa 1988 21 Wrycza S.: Konceptualne modelowanie baz danycch
[22] Wrycza S.: Współczesne metodyki tworzenia systemów informatycznych zarządzania. ORG-SERVICE, Gdańsk 1988
[23] Yourdon E., Constantine L.: Structured Design. Yourdon Press/Prentice-Hall, Englewood Cliffs (NJ), 1979.

meditronik SPÓŁKA z o.o.

UMC UNITED MICROELECTRONICS CORPORATION

HEWLETT PACKARD COMPONENTS

BOURNS

- CZĘŚCI ELEKTRONICZNE
- KOMPUTERY PS/1,PS/2
- DRUKARKI HP
- INSTALACJE SIECI KOMPUTEROWYCH

- UKŁADY PAMIĘCI
- UKŁADY KOMPUTEROWE
- UKŁADY KOMUNIKACYJNE I KOMERCYJNE

- TRANSOPTORY
- WSKAŹNIKI ŚWIETLNE
- WYŚWIETLACZE LED
- PRODUKTY KODÓW KRESKOWYCH
- KONTROLERY I CZUJNIKI RUCHU
- TECHNIKA ŚWIATŁOWODOWA
- ELEMENTY W.CZ. I MIKROFALOWE
- PODZESPOŁY DO MONTAŻU POWIERZCHNIOWEGO (SMD)

- POTENCJOMETRY TRIMPOT
- HYBRYDY REZYSTOROWE
- REZYSTORY SUBMINIATUROWE
- BEZPIECZNIKI MULTIFUSE
- POTENCJOMETRY PRECYZYJNE
- POTENCJOMETRY PANELI CZOŁOWYCH I KODERY
- CEWKI I TRANSFORMATORY
- CZUJNIKI CIŚNIENIA, POŁOŻENIA I PRZYŚPIESZENIA

Partnerzy handlowi: ANALOG DEVICES, ITT, MOTOROLA, SAMSUNG, TELEFUNKEN i inni

PRZEDSTAWICIELSTWO

DYSTRYBUCJA

PMI

DYSTRYBUCJA

IBM Business Partner

meditronik sp. z o.o.

00-194 Warszawa,ul. Dzika 4
tel. (02) 6352263, 6352264
fax (02) 6352195, tlx 816075

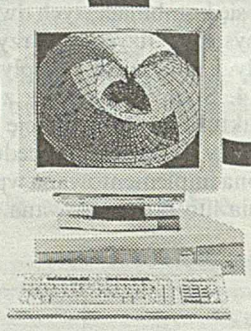
SO/575/91

Liczby przemawiają głośniej, niż słowa.

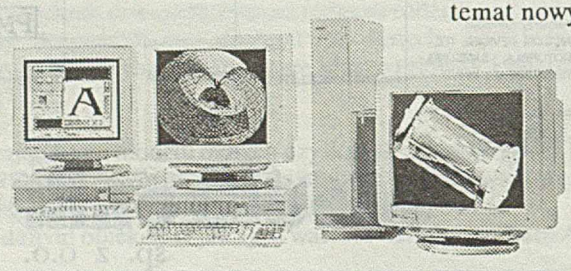
57

57 MIPS/$ 15K

76



76 MIPS/$ 25K



Aby dowiedzieć się więcej na temat nowych stacji roboczych

HP Apollo seria 700 RISC prosimy dzwonić do nas pod numer 37 50 65 i prosić Janusza Rybnika.

HEWLETT PACKARD

HEWLETT-PACKARD Polska sp. z o.o., 01-447 Warszawa, ul. NEWELSKA 6, tel. 37 50 65, tlx 813919, fax 374783 SO/692/91

/unix + dos

Integracja systemów operacyjnych MS-DOS i UNIX (2)

W pierwszej części artykułu (zamieszczonej w poprzednim numerze) Autor omówił emulację terminala znakowego i sieci lokalne w systemie UNIX. Zaczął także naświetlać rolę UNIX-a jako serwera plików dla PC. W bieżącym numerze rozwija ten temat szczegółowo.

● Możliwość tworzenia rozproszonych aplikacji
– do pakietu można dokupić moduł specjalnych bibliotek umożliwiających wykorzystanie mechanizmów komunikacyjnych udostępnianych przez pakiet w programach pisanych przez użytkownika. W przypadku PCI jest to biblioteka dla kompilatora Microsoft C, a w przypadku PC-NFS – funkcje implementujące sposób dostępu do usług sieciowych *sockets* i umożliwiające użycie protokołów RPC i XDR do komunikacji pomiędzy procesami MS-DOS-a i UNIX-a

W wielu pakietach umieszczono sporo dodatkowych mechanizmów ułatwiających pracę użytkowników komputerów osobistych. Najczęściej spotykane to możliwość:
● dostępu do poczty elektronicznej systemu UNIX z systemu MS-DOS,
● tworzenia kopii zapasowych dysków sieciowych i lokalnych dysków komputerów PC na dowolnych urządzeniach (dyskach, napędach taśmowych) zamontowanych na serwerach UNIX-owych,
● inicjowania wykonania programów na serwerach za pomocą komend systemu MS-DOS.

Aby nieco bardziej przybliżyć Czytelnikowi warunki korzystania z komputera osobistego połączonego z serwerem plików, poniżej omówione zostaną podstawowe polecenia wchodzące w skład pakietu PC-Interface.

LOGIN – umożliwia komputerowi PC dostęp do systemu plików maszyny UNIX-owej. W przypadku wykorzystywania karty sieciowej sprawdzane jest, na których komputerach UNIX-owych został zainstalowany serwer PCI, a nazwy sieciowe tych maszyn są wyświetlane jako menu. Oprócz tych nazw wyświetlane są także nazwy portów szeregowych (**COM1**, **COM2** itd.), przez które także można nawiązać sesję z serwerem. Użytkownik wybiera z menu serwer, z którego plików chce korzystać. Następnie musi podać swój identyfikator i hasło; jeśli są one poprawne (jest to sprawdzane przez system operacyjny serwera), to UNIX-owy system plików jest przyłączany jako pierwszy niewykorzystany napęd dysków PC (np. E:) i sterowanie wraca do interpretera poleceń systemu MS-DOS. Jeżeli po rozpoczęciu sesji z serwerem zostanie wykonane polecenie:

DIR E:

to wyświetlony zostanie (oczywiście w formacie MS-DOS-a) UNIX-owy katalog macierzysty danego użytkownika.

LOGOUT – zakończenie sesji i zwolnienie dysku wirtualnego importowanego ze wskazanego jako argument serwera. Bez argumentów polecenie **LOGOUT** odłączą wszystkie dyski wirtualne.

DOS2UNIX

UNIX2DOS – filtry służące do wzajemnej konwersji formatów plików tekstowych: UNIX-a (koniec wiersza – ASCII LF) i MS-DOS-a (koniec wiersza – ASCII CR LF).

ON – uruchomienie polecenia UNIX-a na wskazanym serwerze. Zakończenie wiersza polecenia znakiem „&" powoduje wykonanie polecenia UNIX-owego w tle. Możliwe jest przekierowywanie wejścia i wyjścia i potokowanie programów. Standardowe wyjście polecenia UNIX-owego uruchomionego za pomocą komendy **ON** jest przekształcane do formatu MS-DOS-a (zamiana **CR** na **CR LF**) i wyświetlane na ekranie PD.

Przykład 1

Wykonanie polecenia **ls** na komputerze o nazwie sieciowej **host1** i wyświetlenie jego wyników na ekranie PC:

ON host1 ls

Przykład 2

Wykonanie polecenia **date** na komputerze przyłączonym przez port szeregowy **com1** i zapisanie jego wyników w pliku o nazwie **temp** w bieżącym katalogu dysku wirtualnego reprezentującego system plików tego komputera:

ON com1 „date > temp"

Przykład 3

Wykonanie polecenia **date** na komputerze przyłączonym przez port szeregowy **com1** i zapisanie jego wyników w pliku o nazwie **temp** w bieżącym katalogu bieżącego dysku:

ON com1 date > temp

Jeśli któreś z poleceń UNIX-a jest często używane, to można je wywoływać w skrócony sposób, zmieniając nazwę polecenia **ON**. Np. po wykonaniu polecenia:

COPY ON.EXE LS.EXE

wydanie poleceń:

LS
i

ON – ls

jest równoważne (znak „–" oznacza serwer, z którym jest skojarzony bieżący dysk wirtualny).

PCUPCI – uruchomienie emulatora terminala. Analogicznie do polecenia **LONGIN** należy wybrać maszynę UNIX-ową, której końcówka będzie emulowana. Możliwość emulacji terminala nie jest w żaden sposób uzależniona od tego, czy z danym komputerem UNIX-owym jest otwarta sesja serwera plików.

Sieciowe systemy operacyjne zintegrowane z UNIX-em

W obecnej chwili na rynku komputerów osobistych dominującą rolę w zastosowaniach biurowych odgrywają sieci lokalne i tzw. sieciowe systemy operacyjne NOS (ang. *Network Operating System*). Pozwalają one komputerom typu PC na współdzielenie dysków i drukarek, oferują usługi takie, jak poczta elektroniczna czy serwer bazy danych. Rynek dominują w tej chwili dwa systemy: NetWare firmy Novell oraz LAN Manager firmy Microsoft. Umożliwiają one zarządzanie siecią lokalną mikrokomputerów pracujących w systemach MS-DOS, OS/2 oraz komputerów MacIntosh.

Sieciowe systemy operacyjne są bardzo popularne na świecie, a zintegrowanie ich z wielodostępnymi systemami operacyjnymi wydaje się być naturalnym etapem ich rozwoju. Do niedawna integracja taka była możliwa tylko poprzez specjalne moduły sprzętowe i programowe instalowane na dedykowanych serwerach tych systemów i umożliwiające konwersję protokołów (np. IPX) (SPX<—> TPC/IP), wymianę poczty elektronicznej itp. Moduły takie nie były jednak na tyle uniwersalne, aby stworzyć w tej dziedzinie standard. Ostatnio jednakże został wykonany milowy krok przez samych producentów NOS – zostały zaprojektowane przenaszalne wersje serwerów, które mogą być zainstalowane i pracować równolegle z innym systemem operacyjnym.

W niniejszym artykule zotaną omówione systemy **Portable NetWare** firmy Novell oraz **LAN Manager/X** dostarczany przez firmy AT&T i Microsoft/HP. Należy zaznaczyć, że produkty te nie są dostępne na rynku bezpośrednio od ich twórców. Firmy, które zaprojektowały te pakiety, sprzedają licencję wraz z kodem źródłowym producentom komputerów, którzy sami przystosowują kod pakietów do wymagań systemów operacyjnych swojego sprzętu, a następnie sprzedają „przykrojone" i często ulepszone wersje pod własną nazwą (np. koncern Unisys sprzedaje swoją wersję Portable NetWare pod nazwą **„NetWare for U Series"** firma NCR – pod nazwą **„NCR TOWER NetWare"**). Taki rodzaj sprzedaży nazywa się po angielsku OEM – od słów: *Original Equipment Manufacturing*. Poniżej wymieniono właściwości wspólne dla obu systemów.

Przezroczyste środowisko
Z punktu widzenia stacji roboczych nie ma różnicy pomiędzy typowym serwerem NOS opartym na komputerze typu PC a serwerem NOS zainstalowanym na maszynie UNIX-owej. Oprócz pełnienia funkcji serwera NOS komputer UNIX-owy jest dalej pełnoprawnym systemem wielodostępnym.

Integracja komputera centralnego z sieciami PC
W odróżnieniu od konfiguracji serwera opartej na komputerze typu PC komputer UNIX-owy oprócz pełnienia funkcji serwera plików zapewnia dostęp użytkownikom stacji roboczych do aplikacji UNIX-owych.

Dostęp do sieciowych dysków i drukarek
Użytkownicy stacji roboczych widzą pliki i katalogi znajdujące się na serwerze UNIX-owym dokładnie w takim samym formacie, jaki występuje na serwerach dedykowanych. Tak samo wygląda także dostęp do sieciowych drukarek. Od strony maszyny UNIX-owej pliki eksportowane przez sieciowy system operacyjny są widziane jak wszystkie inne, zwykłe pliki i katalogi UNIX-a. Natomiast system drukarek sieciowych używa bezpośrednio spooler UNIX-a.

Wydzielona administracja sieciowym systemem operacyjnym
Administrowanie serwerem sieciowego systemu operacyjnego jest oddzielone od administrowania macierzystym systemem operacyjnym (UNIX-em). Administrator NOS może zakładać konta użytkowników i przydzielać im zasoby bez potrzeby interwencji administratora systemu podstawowego. Administrowanie sieciowym systemem operacyjnym odbywa się przez terminal maszyny UNIX-owej. Narzędzia administratora, oparte na systemie menu, są łatwe w użyciu.

Tworzenie kopii zapasowych
Tworzenie kopii zapasowych plików serwera jest wykonywane razem z backup-em zwykłych plików UNIX-a – za pomocą klasycznych narzędzi tego systemu.

Interfejs programowy
Pakiet sieciowego systemu operacyjnego jest dostarczany wraz z modułami bibliotecznymi umożliwiającymi tworzenie przez użytkownika własnych programów, które pracując na stacjach roboczych mają dostęp do zasobów systemu operacyjnego UNIX. Przy użyciu tych bibliotek procesy MS-DOS-a mogą komunikować się z procesami UNIX-a. Umożliwia to projektowanie aplikacji o architekturze klient-serwer.

Emulacja terminala UNIX-a na stacjach roboczych
Oprócz dostępu do dysków i drukarek serwera stacje robocze, korzystając z połączenia z siecią lokalną, mogą emulować końcówki komputera UNIX-owego. Wymaga to użycia emulatora terminala, który potrafi wykorzystać sieć lokalną jako swoje łącze komunikacyjne.

Konferencja UNIX-owa

W dniach 4–5 listopada 1991 r. odbyła się w Warszawie konferencja „UNIX – światowy standard dostępu". Konferencję zorganizowała Fundacja Rozwoju Technik Komputerowych we współpracy z Sekcją UNIX-ową PTI. Uczestniczyło ponad 200 osób. Referaty prezentowały problemy techniczne UNIX-a i pewne obszary zastosowań. Artykułem Pawła Żaka rozpoczęliśmy zamieszczanie niektórych materiałów z konferencji w naszym dziale UNIX-owym.

Konferencji towarzyszyła wystawa stacji roboczych z udziałem firm: Bonair, Comp Ltd, Consortia, Digital, EOSystems, Eurocim, Hewlett-Packard, Unisys.

Cechy charakterystyczne dla pakietu Portable NetWare

● Jeżeli do serwera UNIX-owego dołączony jest klasyczny serwer NetWare pracujący na komputerze PC, to wszystkie stacje robocze dołączone do zwykłego serwera przy użyciu obcej dla UNIX-a sieci lokalnej będą także miały dostęp do serwera UNIX-owego. Jest to szczególnie ważne w warunkach polskich, gdzie bardzo wiele instalacji sieci NetWare jest opartych na topologii ARCNet. Zwykle system UNIX nie potrafi obsługiwać adapterów sieciowych tego typu. W tym wypadku, aby dołączyć serwer Portable NetWare do sieci ARCNet, należy do istniejącego dedykowanego serwera (typu PC) przyłączyć adapter Ethernet i skonfigurować tzw. **bridge** – czyli połączyć sieci Ethernet i ARCNet. Po dołączeniu do magistrali Ethernet serwer UNIX-owy będzie dostępny (poprzez bridge) dla wszystkich stacji mających dostęp poprzez sieć ARCNet do bridge-a.
● W obecnej chwili Portable NetWare może używać tylko – typowych dla tego systemu – protokołów IPX/SPX. przewidywana jest w niedalekiej przyszłości możliwość wykorzystania przez ten system protokołów TCP/IP oraz OSI.

Cechy charakterystyczne dla pakietu LAN Manager/X

● W wielu implementacjach pakietu jako interfejs do warstwy transportowej zastosowano szeroko znany standard poziomu sesji o nazwie **NETBIOS.** Jest to standardowy protokół niezależny od sprzętu, używanych w takich produktach sieciowch dla PC, jak **MS-Net, Open Net** i **3Plus.** Po stronie UNIX-a wykorzystuje się interfejs warstwy sesji o nazwie TLI (ang. *Transport Layer Interface*) charakterystyczny dla wersji UNIX System V lub jego odpowiednik pochodzący z implementacji BSD (Berkeley UNIX) o nazwie **sockets.**
● W niektórych implementacjach pakietu przewidziano możliwość zdalnego dostępu do drukarek przyłączonych do stacji roboczych.

cdn

PAWEŁ ŻAK

Ogłoszenie w INFORMATYCE szansą znalezienia właściwego kontrahenta

ul. Racławicka 15/17, 53-149 WROCŁAW
tel./fax (071) 61 54 21, tel. (071) 61 60 61 w. 212
tlx 071 2797

REMECO
ADVANCED COMPUTER SYSTEMS Sp. z o.o.

Dystrybutor

INTERACTIVE
A Kodak Company

wielodostępne wielozadaniowe systemy operacyjne **UNIX** w wersji **3.2** oraz **4.0** z możliwością emulacji systemu **MS DOS** i oprogramowaniem komunikacyjnym (**TCP/IP, NSF**).

SCO
THE SANTA CRUZ OPERATION

INFORMIX®

system zarządzania relacyjną bazą danych zgodny ze standardem **SQL** i oprogramowanie narzędziowe **4GL, SE, SQL** oraz sieciowe **I-NET, I-STAR**

UNIPLEX

zintegrowany wielodostępny system automatyzacji prac biurowych (arkusz kalkulacyjny, edytor tekstów, system zarządzania bazą danych kompatybilną z bazą **Informix**).

Zapewniamy:

☞ *instalacja oprogramowania*
☞ *szkolenie w Polsce*
☞ *realizację zastosowań i sprzedaż oprogramowania użytkowego z wykorzystaniem wyżej wymienionych pakietów*
☞ *doradztwo informatyczne w pełnym zakresie.*

SO/711/91

A jednak Polska uczestniczy w CeBIT'92!

Opublikowana w numerze styczniowym INFORMATYKI zapowiedź Targów CeBIT'92 (str. 7) zawierała stwierdzenie, że Polska już drugi rok nie jest reprezentowana na tej imprezie, a jej udział w 1990 r. ograniczył się tylko do jednej firmy. Była to informacja wyjątkowo deprymująca na tle dalszego stwierdzenia, że inne kraje Europy Środkowej i Wschodniej (Węgry, Czecho-Słowacja, ZSRR), niewątpliwie mniej od nas zaawansowane w reformowaniu gospodarki i rozwoju informatyki, wykazują na tej imprezie coroczny wzrost liczby wystawców i powierzchni ekspozycji. Informacja ta, tak boleśnie drażniąca nasze ambicje narodowe, była całkowicie wiarygodna, pochodziła bowiem z oficjalnych wypowiedzi przedstawicieli Targów (Deutsche Messe AG Hannover) na konferencji prasowej w dniu 8 grudnia ub.r. w Warszawie, a więc już po upływie ostatecznego terminu przyjmowania zgłoszeń wystawców.

Tymczasem w dwa tygodnie później warszawska prasa codzienna zamieściła sensacyjną informację o udziale firmy polskiej w CeBIT'92. W firmie tej dowiedzieliśmy się, że nastąpi to w ramach ekspozycji tematycznej „Software in Europe", ukierunkowanej na promocję dążeń integracyjnych EWG, a w perspektywie – całej Europy. W ramach tej ekspozycji (hala nr 15) odbędzie się konkurs „Software for Europe" mający na celu wyłonienie i uhonorowanie nagrodami „Golden Softies" trzech najlepszych w skali europejskiej oryginalnych produktów programowych. Organizatorem konkursu jest wydawnictwo IDG Communications Verlag AG, które dla każdego zwycięzcy eliminacji z poszczególnych krajów zamówiło i pokryje koszty wynajmu stoiska wystawowego (ok. 15000 DM). Stąd pozorna nieścisłość w oficjalnej informacji organizatorów Targów z 8 grudnia, gdyż zamówienie na stoisko wystawowe reprezentanta naszego kraju pochodzi od firmy niemieckiej.

Jury konkursu powołane przez IDG spośród zgłoszeń producentów oprogramowania z Polski na najlepszy produkt naszego kraju uznało edytor QR-Tekst firmy MALKOM z Warszawy. Ten właśnie produkt będzie reprezentował Polskę w finale konkursu o nagrody „Golden Softies".

QR-Tekst jest obecny na krajowym rynku już czwarty rok. O walorach użytkowych tego edytora świadczy liczba ponad 2000 sprzedanych kopii. Obecnie jest dostarczana znacznie ulepszona wersja 5.1. Oprócz zwycięskiego edytora firma MALCOM przygotowuje na ekspozycję w Hanowerze również inne własne produkty programowe.

Mimo silnej konkurencji kilkunastu krajów, w tym również krajów bardziej od nas zaawansowanych w rozwoju informatyki, można spodziewać się, że dojrzałość, wszechstronność i perfekcja rozwiązań QR-Tekstu stwarza duże szanse dla znalezienia się tego produktu w gronie laureatów „Golden Softies". Mamy nadzieję, że o tym emocjonującym zdarzeniu będziemy mogli poinformować Czytelników w kwietniowym numerze INFORMATYKI.

Praca zbiorowa: Propozycja strategii rozwoju informatyki i jej zastosowań w Rzeczypospolitej Polskiej (2) INFORMATYKA 1992, nr 2, s. 1 Druga część opracowania Polskiego Towarzystwa Informatycznego dla Urzędu Rady Ministrów, zawierającego propozycję strategii rozwoju informatyki i jej zastosowań w Polsce.	**Collective work: Proposal for informatics and its applications development strategy in Poland (2)** INFORMATYKA 1992, No. 2, p. 1 Second part of the Polish Informatics Society's elaboration for the Cabinet, which includes a proposal for informatics and its applications development strategy in Poland.	**Gemeinschaftsarbeit: Ein Vorschlag für die Entwicklung der Informatik und ihrer Anwendungen in Polen (2)** INFORMATYKA 1992, Nr. 2, S. 1 Zweiter Teil von einer Bearbeitung der Polnischen Informatik Gesellschaft für das Ministerrat, die ein Vorschlag für die Entwicklung der Informatik und ihrer Anwendungen in Polen umfasst.
Fryźlewicz Z., Ułasiewicz J.: Scheduler transakcyjny dla zastosowań komunikacyjnych INFORMATYKA 1992, nr 2, s. 10 Charakterystyka rozwiązań oraz sposobu realizacji schedulera transakcyjnego, umożliwiającego zwiększenie szybkości transmisji danych.	**Fryźlewicz Z., Ułasiewicz J.: Transaction scheduler for communication applications** INFORMATYKA 1992, No, 2, p. 10 Characteristics of solutions and realization method of the transaction scheduler, which enables to increase the data transmission speed,	**Fryźlewicz Z., Ułasiewicz J.: Transaktionsscheduler für Kommunikationsanwendungen** INFORMATYKA 1992 Nr, 2, S. 10 Eine Charakteristik von Lösungen und Realisationsmethode eines Transaktionsschedulers, der eine Geschwindigkeitssteigerung der Datenübertragung ermöglicht.
Zonenberg D.: Emulator układowy mikroprocesorów jako narzędzie uruchomieniowe w ujęciu programistycznym INFORMATYKA 1992, nr 2, s. 15 Próba zdefiniowana funkcji uniwersalnego emulatora układowego mikroprocesorów za pomocą języka Pascal.	**Zonenberg D.: Hardware emulator for microprocessors as a development tool in programming formulation** INFORMATYKA 1992, No. 2, p. 15 An attempt for defining functions of universal hardware emulator for microprocessors using the Pascal language.	**Zonenberg D.: Hardwareemulator für Mikroprozessoren als ein Entwicklungswerkzeug in Programmierungsfassung** INFORMATYKA 1992, Nr. 2, S. 15 Ein Definierungsversuch von Funktionen eines universalen Hardwareemulators für Mikroprozessoren mit Hilfe der Pascal-Sprache.
Cupisz R.: LINC – kompletne środowisko CASE INFORMATYKA 1992, nr 2, s. 20 Charakterystyka właściwości systemu LINC, produktu firmy UNISYS przeznaczonego do projektowania dużych systemów informatycznych, który łączy w sobie cechy narzędzia typu CASE oraz języków czwartej generacji.	**Cupisz R.: LINC – a complete CASE environment** INFORMATYKA 1992, No. 2, p. 20 Characteristics of the LINC system features, a product of UNISYS for big data processing system designing, which joins the attributes of a CASE tool and of a fourth generation language.	**Cupisz R.: LINC – eine vollständige CASE--Umwelt** INFORMATYKA 1992, Nr. 2, S. 20 Eine Charakteristik von Eigenschaften des LINC-Systems, eines UNISYS-Produktes für Projektierung von grossen EDV-Systemen, das die Merkmale eines CASE-Werkzeuges und einer 4. Generation-Sprache vereinigt.
Stąpor K.: Strukturalna analiza i projektowanie systemów informatycznych – metodyka Gane'a – Sarsona (1) INFORMATYKA 1992, nr 2, s. 24 Pierwsza część charakterystyki metody Gane'a – Sarsona, stosowanej do strukturalnej analizy oraz projektowania systemów informatycznych.	**Stąpor K.: Structural analysis and data processing system designing – the Gane-Sarson method (1)** INFORMATYKA 1992, No. 2, p. 24 First part of characteristics of the Gane-Sarson method, which is applied for structural analysis and data processing system designing.	**Stąpor K.: Strukturanalyse und EDV-Systeme Projektierung – die Gane-Sarson-Methode (1)** INFORMATYKA 1992, Nr. 2, S. 24 Erster Teil einer Charakteristik von Gane-Sarson-Methode, die zur Strukturanalyse und EDV-Systeme Projektierung verwendet wurde.

Szanowna Redakcjo!

Gratuluję Redakcji realizacji pomysłu otwarcia stałej rubryki /unix. Chcąc-niechcąc, praktyczne uprawianie informatyki w Polsce niegdyś powiatowej, polega na korzystaniu z kopii mikrokomputerów IBM PC/XT/AT z systemem operacyjnym MS-DOS. Relatywnie niska cena sprzętu, powszechne – niestety – korzystanie z pirackich kopii oprogramowania użytkowego niejako przywiązało użytkowników do jednostanowiskowego, jednozadaniowego i ograniczonego do ok. 612 KB pamięci operacyjnej, systemu. Różne sztuczki, czyli programy sterujące dostępem do pamięci operacyjnej powyżej 640 KB przez MS-DOS, w rodzaju Desqview czy NetRoom, praktycznie dzisiaj nie są w Polsce powszechnie używane. Może tylko środowisko oferowane przez MS Windows 3.0 wzbudziło poważniejsze zainteresowanie, jako że jest to szansa na definitywne zamknięcie tzw. „problemu polskich liter". Ja ze swej strony przekonałem się o pewnych istotnych szczegółach – Windows 3.0 pracuje ze znośną dla użytkownika wydajnością, gdy mają do dyspozycji co najmniej 2 MB pamięci RAM, szybki twardy dysk rzędu 65 MB pojemności oraz procesor i80386DX-20 MHz.

Mimo, że systemy operacyjne wywodzące się od UNIX-a oferują zasadniczo doskonalsze środowisko pracy, zwłaszcza dla zaawansowanych informatyków, systemy te nadal są czymś bardzo tajemniczym i mało dostępnym na polskiej prowincji. Zasadniczą przyczyną tego stanu, jak mi się wydaje – są ceny produktów unixowych, niezależnie od sprzętu na którym będą pracować. Oferowane przez Microsoft dla właścicieli IBM PC oprogramowanie kosztuje około 500 dolarów za dowolny kompilator czy procesor tekstu pracujący w systemie Xenix lub SCO UNIX. Arkusz kalkulacyjny Lotus 1-2-3 pracujący w systemie SCO UNIX kosztuje, jak na razie, co najmniej 1000 dolarów; komplet SCO UNIX (Operating + Develop) jest sprzedawany za około 1450 dolarów; unixowa wersja SCO FoxBase + 386 kosztuje około 720 dolarów. Jeśli chcemy pracować w systemie QNX 4.0 musimy wydać około 1100–1150 dolarów na każde stanowisko w sieci (system + specjalny sprzęt sieciowy) oprócz kosztu mikrokomputera typu AT.

Nasuwa się więc pytanie, czy bariera cenowa jest do pokonania w obecnej polskiej rzeczywistości? Wydaje mi się, że warto zainteresować się systemem operacyjnym COHERENT. Tak zwany „pierwszy rzut oka" określi ten system jako idealną kopię systemu UNIX V. 7, z roku 1979 i programem shell napisanym przez Bourne'a. Oczywiście zawiera wiele ulepszeń włączanych do UNIX-ów od 1979 roku, choćby edytor typu vi lub możliwość czytania i zapisywania plików w formacie wymaganym przez MS-DOS. COHERENT r. 3.2 v2 jest w pełni kompletnym i niezależnym systemem operacyjnym, wielodostępnym i wielozadaniowym.

System operacyjny COHERENT jest produktem stworzonym i sprzedawanym wyłącznie przez firmę Mark Williams Company. COHERENT jest prawnie zastrzeżonym znakiem towarowym tej firmy. System ma mechanizmy ochrony pamięci, asembler procesora 8086, kompletny kompilator języka C, kilka edytorów tekstowych, programy: awk, lex, yacc, make, UUCP oraz większość znanych programów narzędziowych systemów typu UNIX. Wymagania sprzętowe COHERENT-a są skromne – wystarczy mikrokomputer typu IBM AT. Przechodząc do szczegółów: procesor typu i80286, nowszy; minimum 640 KB pamięci RAM, jeden napęd dyskowy (dysk A:) 1,2 MB albo 1,44 MB, konieczny jest dysk twardy o pojemności minimum 20 MB; monitor + karta typu Hercules albo EGA, a także VGA. Pełne zainstalowanie systemu (pięć gier komputerowych również!) zajmuje około 7,2 MB obszaru twardego dysku. Jeśli komputer typu IBM AT ma zainstalowany 1 MB pamięci RAM, to po włączaniu system zgłasza 844 KB wolnej pamięci operacyjnej. Oczywiście są i dziury w całym. Pod kontrolą COHERENT-a nie można wykonywać programów pracujących w systemie MS-DOS, brakuje graficznego trybu pracy monitorów, jest dostępny jedynie tryb tekstowy 80 × 25. COHERENT nie pracuje na maszynach typu PS/2 (Micro Channel Architecture). Najnowsze typy sterowników twardych dysków, zwłaszcza budowane w stale zmieniającym się standardzie SCSI, mogą wymagać od użytkownika samodzielnego napisania programowego sterownika, czyli device drivera. Za dodatkową opłatą 40 dolarów użytkownik otrzymuje od producenta oprogramowanie ułatwiające to zadanie – Device Driver Kit. Stworzenie sieci w zasadzie jest możliwe przy wykorzystaniu istniejących portów szeregowych COM1...4 i ewentualnie dodatkowych kart zwiększających (do 8) liczbę portów szeregowych.

Ponieważ obecnie sam jeszcze raczkuję w obszarze systemu COHERENT, przytoczę opinię pana Davida Fiedlera, zamieszczoną w czasopiśmie „BYTE" z listopada 1990 roku:

Jeśli chcesz nauczyć się systemu UNIX i przy jego pomocy tworzyć nowe oprogramowanie, a przy tym wydając mniej niż sto dolarów, nie widzę przeciwskazań abyś wybrał Coherent.

Pozytywną opinię o systemie operacyjnym COHERENT wyraził także pan Sean Fulton w magazynie „UNIX Today!" z 26 listopada 1990 roku:

Firma Mark Williams Company doskonale opanowała sztukę imitacji, system operacyjny Coherent można zakwalifikować jako bliźniaczą kopię UNIX-a zresztą pracując z nim sam stwierdzisz – Nie mogę uwierzyć, że to nie jest UNIX.

Elbląg, 1991.12.03

Z poważaniem
JACEK PRZEPIÓRA

Ogłoszenie w INFORMATYCE
dotrze przede wszystkim do profesjonalistów